高等学校公共课计算机规划教材

C程序设计与应用教程题解及实验

蔡启先　刘永娟　何春华　等编著

http://www.phei.com.cn

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

高等学校公共课计算机规划教材

C 程序设计与应用教程题解及实验

蔡启先　刘永娟　何春华　等编著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书是《C程序设计及应用教程》（电子工业出版社出版）一书配套的习题解答与实验教材。内容包括：《C 程序设计及应用教程》各章的题解、结合新的全国计算机等级考试要求补充的练习题、Visual C++/Turbo C 上机指导、常见上机错误与纠错方法，以及与教材同步的 C 程序设计实验，附录提供了补充练习题答案等相关资料。书中提供的全部程序都按照结构化程序设计方法采用统一的缩格方式编写，并经上机验证。为便于读者自学，程序中添加了大量注释。

本书可独立地作为学习 C 语言和实践上机的必备参考书，可作为高等学校各专业、计算机水平考试、各类成人教育的教材，也可作为从事计算机应用的科技人员的参考书和培训教材。

图书在版编目 (CIP) 数据

C 程序设计与应用教程题解及实验/ 蔡启先等编著．—北京：电子工业出版社，2009.1
高等学校公共课计算机规划教材
ISBN 978-7-121-07923-8

Ⅰ．C… Ⅱ．蔡… Ⅲ．C 语言－程序设计－高等学校－教学参考资料 Ⅳ．TP312

中国版本图书馆 CIP 数据核字（2008）第 188367 号

策划编辑：冯小贝
责任编辑：余 义
印 刷：
装 订：
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
开 本：787×1092 1/16 印张：15.5 字数：397 千字
印 次：2009 年 1 月第 1 次印刷
印 数：4000 册 定价：25.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010)88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：(010)88258888。

前　言

C 语言是国内外广泛流行的计算机高级程序设计语言。目前，C 语言程序设计课程不仅是计算机科学与技术专业的必修专业基础课，而且已成为非计算机专业的一般计算机应用人员学习计算机高级语言的首选课程。该课程实践性强，课外练习和上机训练尤为重要。为适应高等学校学生及广大计算机爱好者和应用人员的需要，我们曾编写了《C 程序设计及应用教程》（电子工业出版社出版），为了更好地发挥教材的作用，充分调动学生自主学习的积极性，特编写本书。

全书分为 4 大部分。第 1 部分“习题题解”中，对主教材中各章后面的编程练习题均给出解答，并在程序清单中列出必要的注释，以便读者在独立解题后进行参考。众所周知，解题的算法是多种多样的，因此本书中给出的解答并非唯一，读者可从中悟出思路，获得启发。第 2 部分精心编写了“补充练习题”。补充练习题在内容取舍、题型上参照了全国计算机等级考试的要求，结合了高等学校的教学实际，为读者巩固 C 语言基础知识，加强编程的基本训练和参加计算机等级考试进行了准备。补充练习题的参考答案列于附录 B 中。第 3 部分是“上机指导”，详细介绍了以 Visual C++为主开发 C 程序的实践，同时兼顾了传统的 Turbo C 操作，并根据作者长期的教学经验，总结分析了一些常见的错误类型与纠错方法，分类提供给读者参考。读者可依据这部分的知识顺利完成 C 程序的上机编程和调试。第 4 部分是“C 语言程序设计实验”，这是配合主教材教学而提供的上机实验教材。每一个实验均有“实验目的”、“实验准备”、“实验内容”和“实验报告要求”，都与教材的相应章节内容同步，既方便学生自主准备实验和独立上机操作，也便于教师结合教材安排和指导学生实验。本书附录提供的相关资料便于读者上机时查阅了解出错信息，以及了解全国计算机等级考试的要求。书中提供的全部程序都按照结构化程序设计方法采用统一的缩格方式编写，并经上机验证。

全书由蔡启先、刘永娟、何春华、刘智和刘智琦等编著。由蔡启先教授担任主编著，刘永娟、何春华担任副主编著。其中，第 1 部分由蔡启先（第 1 章习题题解）、何春华（第 2，3 章习题题解）、朱亚超（第 4，5，7 章习题题解）、刘永娟（第 6，8 章习题题解）、刘琦（第 9，10 章习题题解）编写，第 2 部分和附录 B 由刘永娟和刘智共同编写，第 3 部分和附录 A、附录 C 由蔡启先编写，第 4 部分实验由何春华、刘智琦共同编写，最后由蔡启先统稿。在全书编写过程中，得到了作者所在单位广西工学院的大力协助，电子工业出版社责任编辑对本书的编写和出版自始至终进行了非常宝贵的指导和支持，谨在此一并致谢。在本书编写过程中，借鉴了若干出版物，也在此向有关作者表示感谢。

由于编写时间仓促，错误在所难免，敬祈专家和读者指正。

作　者

2008 年 10 月

目　　录

第 1 部分　习 题 题 解

第 1 章　C程序设计概述

1.1　利用 printf()编写一个输出下面信息的 C 程序：

```
**************************
 One World, One Dream
**************************
```

解：

```
#include <stdio.h>
void main()
{   printf("**************************\n");
    printf("One World, One Dream\n");
    printf("**************************\n");
}
```

1.2　编写一个 C 程序，输入圆半径，求圆周长和圆面积。

解：

```
#include <stdio.h>
void main()
{   int  r;              /* 设半径 r 为整型 (int 型) 变量  */
    float  area;         /* 设 area 为实型(float 型)变量  */
/* 提示输入圆半径 r，末尾不换行 */
    printf("Input r:");
/* 调用输入函数 scanf()，从键盘输入一个数据给 r */
    scanf("%d",&r);
/* 计算出圆面积，将结果赋给变量 area */
    area=3.14159*r*r;
/* 输出结果，显示 area=字样，\n 表示换行 */
    printf("area=%f\n",area);
}
```

1.3　求 s＝1＋3＋5＋…＋99。写出算法，画出流程图，编写 C 程序。

解：下面是具体的算法（流程图如图 1.1 所示）：

① s←0；（累加器 s 赋初值 0）

② i←1；（i 中存放要累加的第一个数）

③ 将 i 和 s 的值相加，即 i+s→s；（这个过程完成了累加一个数的操作）

④ 使 i 的值加 2，即 i+2→i；（计算要累加的下一个数）

⑤ 若 i≤99 则返回③继续执行，否则执行⑥；（i≤99 应返回③继续累加 i，否则，已累加完，应执行步骤⑥）

⑥ 打印出 s 的值。

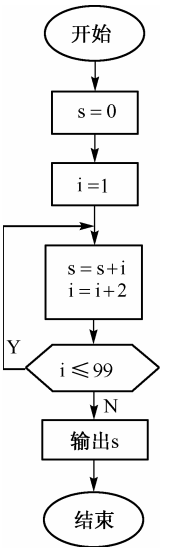

图 1.1　习题 1.3 的算法流程图

下面是程序：

```
#include <stdio.h>
```

```
void main()
{   int  s,i;                  /* 设 s,i 为整型变量  */
    s=0; i=1;                  /* 循环前将 s 清零, i 存放要累加的初值 */
    do                         /* do{…}while 为循环控制语句 */
    {   s=s+i;                 /* 累加 */
        i=i+2;                 /* 计算要累加的下一个数 */
    }while(i<=99);             /* 循环条件判断 */
    printf("sum=%d\n",s);      /* 输出结果，显示 sum=字样，\n 表示回车 */
    }
```

1.4　有 8 只乒乓球，其中一只重量较小，其他的质量相同。若用天平来找出较小的那只球，能设计出几种找法？指出哪一种找法在天平上称的次数最少？画出次数最少的找法的操作流程图。

解：共有 4 种找法。

找法 1（平均称 4 次）：

天平一端放 1 只乒乓球，其余 7 个球依次放天平另一端进行比较，最快只比较 1 次，最慢要比较 7 次，平均比较次数为(7+1)/2=4 次。

找法 2（平均称 2.5 次）：

将 8 只乒乓球分为 4 组，每组 2 个，分组依次比较。其比较次数最少 1 次，最多 4 次，平均比较次数为(1+4)/2=2.5 次。

找法 3（称 3 次）：

① 8 只乒乓球分为 2 组，每组 4 个。在天平上比较后，找出较轻的一组。

② 将较轻的一组又分为 2 组，每组 2 个。在天平上比较后，找出较轻的一组。

③ 将较轻的一组又分为 2 组，每组 1 个。在天平上比较后，最后找到较轻的那只球。

找法 4（称 2 次）：

① 8 只乒乓球分为 3 组，头两组每组 3 个，第三组为 2 个。在天平上比较头两个组后，有两种情况：(a) 如果相等，则较轻的必在第三组中。(b) 如果不等，则较轻的必在头两组中。

② (a) 如果头两组相等，将第三组的 2 个球在天平上比较后，最后找到较轻的那只球。(b) 如果头两组不等，从较轻的一组球中任意取 2 个球，在天平上进行比较，若相等，则较轻的球必是该组中未比较的那个球；若不等，则较轻的那只球最后找到。

显然，找法 4 次数最少，设 8 只乒乓球分别为 p1，p2，p3，p4，p5，p6，p7，p8，找法 4 的操作流程图如图 1.2 所示。

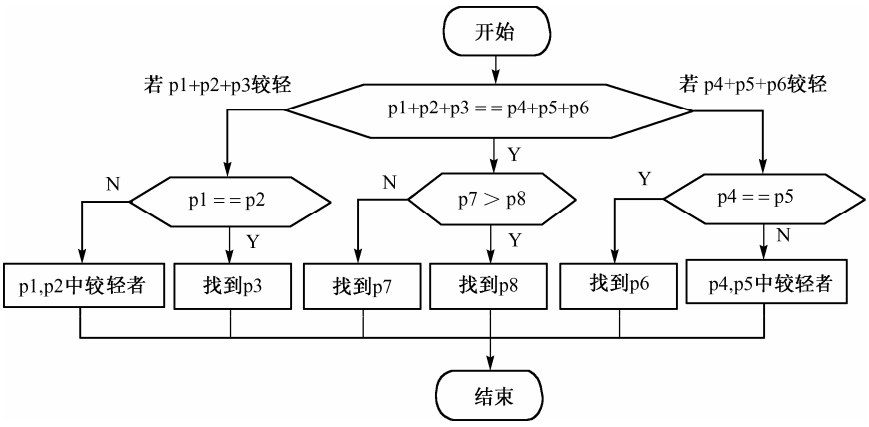

图 1.2　习题 1.4 找法 4 的操作流程图

1.5　自定义一个函数，能将大写字符变为小写字符。利用主函数完成输入/输出字符的功能，每输入一个字符后调用自定义函数，直至完成 10 个字符的处理。写出算法，画出流程图，并编写 C 程序。

解：算法与流程图如图 1.3 所示。

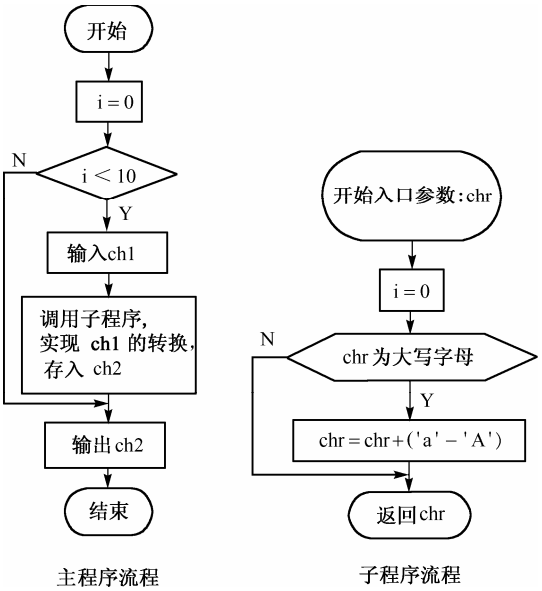

图 1.3　习题 1.5 的算法流程图

C 程序如下：

```
#include <stdio.h>
void main()
{   char  ch1, ch2;              /* 设置字符型变量 ch1, ch2 */
    char  toupper(char chr);     /* 声明本函数要调用 char 型函数 toupper，有 char
                                    型调用参数 */
    for(i=0;i<10;i=i+1)          /* 循环 10 次 */
    {   scanf("%c", &ch1 );      /* 输入 1 个字符 */
        ch2= toupper(ch1);       /* 调用 toupper 函数，实现 ch1 的转换，存入 ch2 */
        printf("%c\n", ch2);     /* 输出转换后的 1 个字符 */
    }
}
char toupper(char chr)
{   if(chr>= 'A' && chr<='Z')    /* 判断 chr 是否为大写字母 */
        chr=chr+('a'-'A');       /* 若 chr 为大写字母，则变为小写字母 */
    retern(chr);                 /* 函数返回转换后的字符 */
}
```

第 2 章　数据类型和表达式

2.1　下列哪些是 C 语言合法的常量？ 每个常量是整型、字符型还是浮点型？

(1) 077　　(2) 0x77　　(3) 79.　　(4) 079　　(5) .204　　(6)　'\101'

(7) 12.E3.5　　(8) e5　　(9) 'ab'　　(10) '\x41'　　(11) 0101

解：整型常量：(1) (2) (11)；字符型常量：(6) (10)；浮点型常量：(3) (5)。

2.2 求下列表达式的值。

(1) a+(int)b%2*(int)(a+b)%5/7，其中 a = 6.3，b = 3.6。

(2) (float)(x+y)/3+(int)y%(int)x，其中 x = 2，y = 5。

(3) a – = 2，a + = a，a *= 2+3，其中 a = 15。

解：(1) 6.300000　　(2) 3.333333　　(3) 130

2.3 下面语句的输出是什么?

(1) `printf("%c\n",'a');`　　(2) `printf("%d\n",'a');`

(3) `putchar('\141');`　　(4) `putchar('\x61');`

解：(1) a　　(2) 97　　(3) a　　(4) a

2.4 写出下列程序的运行结果：

(1)
```
#include <stdio.h>
void main()
{   int  x=3 ,y=5;
    printf("%d,%d,%d,%d",x/y,x%y,y/x,y%x);
}
```

(2)
```
#include <stdio.h>
void main()
{   int i=7,j=8;
    i*=j+1;i+=j+=2+3;
    printf("%d,%d",i,j);
}
```

(3)
```
#include <stdio.h>
void main()
{   char c1,c2;
    int i,j;
    c1='a'; c2='b';
    printf("c1=%c\t%d\t%o\t%x\n",c1,c1,c1,c1);
    printf("c2=%c\t%d\t%o\t%x\n",c2,c2,c2,c2);
}
```

(4)
```
#include <stdio.h>
void main()
{   int  x=4,y=20;float a=4.4,b=5.5;
    printf("%f",(float)(x+y)/5+(int)b%(int)a);
}
```

(5)
```
#include <stdio.h>
void main()
{   int i,j,m,n,s;
    i=5; j=10;
    m=++i;
    n=j++;
    s=m++-n;
    printf("i=%d\tj=%d\tm=%d\tn=%d\ts=%d\n",i,j,m,n,s);
}
```

解：(1) 0, 3, 1, 2

(2) 76, 13

(3) c1 = a　97　141　61

c2 = b　98　142　62

(4) 5.800000

(5) i = 6　　j = 11　　m = 7　　n = 10　　s = – 4

2.5　下列每段代码的输出结果是什么？假设其中的 i、j 和 k 都是 int 型的变量。

(1) `i = 5;printf("%d\n",i++–5);printf("%d",i);`

(2) `i = 5;printf("%d\n",++i–5);printf("%d",i);`

(3) `i = 7;j = 8;k = 9;printf("%d\n",i+++j++ – (––k));printf("%d,%d,%d",i,j,k);`

解：(1) 0　　(2) 1　　(3) 7

6　　6　　8,9,10

2.6　摄氏温度（用 c 表示）与华氏温度（用 f 表示）的关系式为

$$c = \frac{5}{9}(f - 32)$$

编写一个程序，将华氏温度转换成摄氏温度：求华氏温度 f 为 60，90，120 时的摄氏温度。所得程序要能显示下面的运行示例：

```
Please input the fahrenheit scale:60↙
the centigrade is 15.56
```

（注意，分数 5/9 应写成 5.0/9，如果还是写成 5/9，会产生什么结果？还要注意 f 变量要声明为 float 型）

解：
```
#include <stdio.h>
void main()
{  float  f,c;
   printf("Please input the fahrenheit scale:");
   scanf("%f",&f);
   c=5.0/9*(f-32);
   printf("the centigrade is %.2f",c);
}
```

程序运行情况如下：

```
Please input the fahrenheit scale:100↙
the centigrade is 37.78
```

2.7　编写一程序，输入两个整数 a 和 b，交换这两个整数，输出交换后的结果。

解：
```
#include <stdio.h>
void main()
{  int a,b,t;
   printf("Please input two integer: ");
   scanf("%d%d",&a,&b);
   t=a;a=b;b=t;
   printf("After swap a=%d,b=%d",a,b);
}
```

程序运行情况如下：

```
Please input two integer:50 70↙
After swap a=70,b=50
```

2.8　编写一程序，输入三角形的三条边 a、b、c，求三角形面积。要求保留两位小数，对第三位小数四舍五入。

解：已知三角形三边长 a、b、c，求三角形面积可用海伦公式：

三角形面积 $\text{area}=\sqrt{s(s-a)(s-b)(s-c)}$，其中 $s=\dfrac{a+b+c}{2}$，下面是本题程序：

```
#include <stdio.h>
#include <math.h>
void main()
{   int a,b,c;
    float s,;
    printf("Please input tree edges of the triangle:");
    scanf("%d%d%d",&a,&b,&c);
    s=(a+b+c)/2;
    area=sqrt(s*(s-a) *(s-b) *(s-c));
    printf("the area is %.2f",area);
}
```

程序运行情况如下：

```
Please input tree edges of the triangle:5 6 7↙
the area is 14.70
```

2.9 编写一程序，计算 $y=\dfrac{5x^3\sqrt{3.2x+5.6}}{9.3x-10.7}$，x 的值从键盘输入。

解：

```
#include <stdio.h>
#include <math.h>
void main()
{  float x,y;
   printf("Please input x:");
   scanf("%f",&x);
   y=5*x*x*x*sqrt(3.2*x+5.6)/(9.3*x-10.7);
   printf("y=%f",y);
}
```

程序运行情况如下：

```
Please input x:4.2↙
y=56.996092
```

2.10 编程显示 sizeof(int)、sizeof(char)、sizeof(float)、sizeof(double)、sizeof(long)的值。

解：

```
#include <stdio.h>
void main()
{  printf("sizeof(int)=%d\n",sizeof(int));
   printf("sizeof(char)=%d\n",sizeof(char));
   printf("sizeof(float)=%d\n",sizeof(float));
   printf("sizeof(double)=%d\n",sizeof(double));
   printf("sizeof(long)=%d\n",sizeof(long));
}
```

程序运行情况如下：

```
sizeof(int)=4
sizeof(char)=1
sizeof(float)=4
sizeof(double)=8
sizeof(long)=4
```

本题程序可用来检查你所用 C 编译系统中的数据类型长度。

第 3 章　算法的基本控制结构

3.1　下列程序的输出结果是什么？

(1)
```
#include <stdio.h>
void main()
{  int  a=1,b=20,c=10;
   printf("a<b<c=%d",a<b<c);
}
```

(2)
```
#include <stdio.h>
void main()
{  int  a,b,k;
   k=(a=2,b=5,a>b?a++:b++,a+b);
   printf("a=%d,b=%d,k=%d",a,b,k);
}
```

解：(1) a<b<c=1

(2) a＝2，b＝6，k＝8

3.2　假设 x＝1,y＝3,z＝2,u,k;求下列表达式的值。

(1) `u = --x&&y++||z++;`

(2) `k = x<y<z&&z||z/y;`

解：(1) u=1　　(2) k=1

3.3　假设 i,j,k 都是 int 型的变量，下列语句片段的输出结果是什么？

(1)
```
i=10;j=5;
printf("%d",!i<j);
```

(2)
```
i=3;j=4;k=5;
printf("%d",i%j+i<k);
```

(3)
```
i=7;j=8;k=9;
printf("%d",i-7&&j++||k++);
printf("%d,%d,%d",i,j,k);
```

解：(1) 1　　(2) 0　　(3) 1 7，8，10

3.4　编写一个程序，输入三角形的三条边 a、b、c，求三角形面积。要求保留两位小数，对第三位小数四舍五入。注意本题与练习题 2.8 不同，它要求程序设计必须严密健壮。如果输入的三角形的三条边的数据不满足构成三角形的条件（即任意两边之和大于第三边），则必须重新输入三条边的数据。

解：
```
#include <stdio.h>
#include <math.h>
void main()
{  int a,b,c;
   float s,area;
   do
   {   printf("Please input tree edges of the triangle:");
       scanf("%d%d%d",&a,&b,&c);
       if((a+b)>c && (a+c)>b && (b+c)>a)  /* 任意两边之和大于第三边 */
           break;                         /* 退出循环 */
       else
           printf("Error!!!,try again\n");
   }while(1);
```

```
    s=(a+b+c)/2;
    area=sqrt(s*(s-a)*(s-b)*(s-c));          /* 已知三条边求面积 */
    printf("the area is %.2f",area);
}
```

程序运行情况如下：

```
Please input tree edges of the triangle:10 5 4↙
Error!!!,try again
Please input tree edges of the triangle:5 6 7↙
the area is 14.7
```

3.5 下面程序段的功能是什么？结果是多少？

```
sum=0;
for(i=0;i<10;i++)
{  if(i%2) continue;        /* 如果 i 是奇数，则不累加 */
   sum+=i;
}
printf("%d\n",sum);
```

解：程序实现求 0～9 之间的偶数之和，奇数将不累加。结果是 20。

3.6 输入一个不超过 5 位数的数，将该数反序输出。例如，数 12345，反序输出 54321。

解：

```
#include <stdio.h>
#include <math.h>
void main()
{  int number,digit;
   printf("Please input one integer:");
   scanf("%d",&number);
   printf("the reversal is :");
   do
   {   digit=number%10;          /* 取得 number 的个位数 */
       printf("%d",digit);
       number=number/10;         /* 去掉 number 的个位，将 number 缩小 10 倍 */
   }while(number>0);
}
```

程序运行情况如下：

```
Please input one integer: 4287↙
the reversal is :7824
```

3.7 打印出所有的“水仙花数”。所谓“水仙花数”是指一个三位数，且其各位数字的立方和等于该数本身。例如，153 是一个水仙花数，因为 $153=1^3+5^3+3^3$。

解：

```
#include <stdio.h>
void main()
{  int flower,i,j,k;                  /* i 为百位数，j 为十位数，k 为个位数 */
   for(i=1;i<=9;i++)
       for(j=0;j<9;j++)
           for(k=0;k<9;k++)
               {   flower=i*100+j*10+k;
                   if(flower==i*i*i+j*j*j+k*k*k)
                       printf("%5d",flower);
               }
   printf("\n");
}
```

程序运行情况如下：

```
153 370 371 407
```

3.8　输入一行字符（以回车键结束），分别统计其中的英文字母、空格、数字和其他字符的个数。

解法 1：根据字符的 ASCII 码值进行比较判断，分类进行计数。

```
#include <stdio.h>
#include <ctype.h>
void main()
{  int lcount,dcount,scount,others;
   char ch;
   lcount=dcount=scount=others=0;
   while((ch=getchar())!='\n')/*从终端读取一个字符存于 ch，该字符不是回车键 */
   if(ch>='A' && ch<='Z' || ch>='a' &&ch<='z') lcount++;
                                                    /* ch 是字母 */
   else if(ch>='0' && ch<='9') dcount++;            /* ch 是数字字符 */
       else if(ch==' ' || ch=='\t') scount++;       /* ch 是空白字符 */
           else others++;
   printf("letter=%d,digit=%d,space=%d,others=%d",lcount,dcount,
       scount, others);
}
```

程序运行情况如下：

```
china*CHINESE**666666  **888888↙
letter=12,digit=12,space=2,others=5
```

解法 2：直接调用字符判断函数进行分类计数。

```
#include <stdio.h>
#include <ctype.h>              /* 相关字符判断函数由 ctype.h 文件定义 */
void main()
{  int lcount,dcount,scount,others;
   char ch;
   lcount=dcount=scount=others=0;
   while((ch=getchar())!='\n')
   if(isalpha(ch)) lcount++;                /* 对字母计数 */
   else if(isdigit(ch)) dcount++;           /* 对数字计数 */
        else if(isspace(ch)) scount++;      /* 对空格计数 */
           else others++;
   printf("letter=%d,digit=%d,space=%d,others=%d",lcount,dcount,
     scount,others);
}
```

程序运行情况同解法 1。

3.9　计算 1 到 200 之间的奇数之和及偶数之和。

解：

```
#include <stdio.h>
#include <ctype.h>
void main()
{  int i,evensum,oddsum;    /* oddsum 为奇数之和，evensum 为偶数之和 */
   evensum=oddsum=0;
   for(i=1;i<=200;i++)
       if(i%2==1) oddsum+=i;
       else evensum+=i;
```

```
    printf("oddsum=%d,evensum=%d",oddsum,evensum);
}
```

程序运行情况如下：

```
oddsum=10000,evensum=10100
```

3.10 输出 100 以内能被 3 整除且个位数为 6 的所有整数。

解：

```
#include <stdio.h>
#include <ctype.h>
void main()
{  int i;
   for(i=6;i<=100;i+=10)                    /* i+=10 保证个位数为 6 */
       if(i%3==0) printf("%5d",i);
}
```

程序运行情况如下：

```
 6 36 66 96
```

3.11 求 $s=1^2+2^2+3^2+\cdots+n^2\leqslant 1000$ 时的最大 n。

解：

```
#include <stdio.h>
#include <ctype.h>
void main()
{  int i=0,pre,s=0;
   while(1)
   {   s+=i*i;              /* 累加 i 的平方 */
       if(s>1000) break;
       pre=i;               /* pre 记下当前累加和小于 1000 时的 i */
       i++;
   }
   printf("%5d",pre);
}
```

程序运行情况如下：

```
13
```

第 4 章 函 数

4.1 编写函数求两个整数中较大的数，用主调函数调用这个函数求出 n 个整数中的最大数，并输出。要求 n 个整数由键盘输入。

解：

```
#include <stdio.h>
int Bigger(int x, int y);
void main()
{   int i;                                  /* 输入整数个数计数 */
    int n=5;                                /* 要输入的整数个数 */
    int number;                             /* 临时存储输入的整数 */
    int biggest;                            /* 存储最大整数 */
    printf("请输入 %d 个整数:\n", n);
    scanf("%d", &biggest);                  /* 初始化 biggest */
    for (i=0; i<n-1;i++)
    {   scanf("%d", &number);               /* 输入剩余的 n-1 个整数 */
        biggest = Bigger(biggest, number);
                                    /* 调用 Bigger 函数求两个整数中较大的数 */
```

```
        }
        printf(" %d 个数中最大的数是 %d \n", n, biggest);
    }
    int Bigger(int x, int y)
    {   if (x>=y)
            return (x);
        else
            return (y);
    }
```

程序运行情况如下：

```
请输入 5 个整数:
15  -24  10  43  -61↙
5 个数中最大的数是 43
```

4.2　编写一个函数，输入一个 4 位整数，输出该 4 位数的 4 个数字，并求其和。

解：

```
#include <stdio.h>
void DisplayAndSum(int num);
void main()
{  int num;
   printf("请输入一个 4 位整数：");
   scanf("%d", &num);
   DisplayAndSum(num);     /* 输出该数的数字并计算这些数字的和 */
}
/* 该程序使用于任意个数的整数 */
void DisplayAndSum(int num)
{  int remainder;
   int sum=0;
   if (num<0) num=-num;    /* 将输入数据转换为非负整数，然后求解其各位数据 */
   printf("该数的各位数字,从个位开始依次为");
   while (num!=0)
   {   remainder=num%10;
       printf("%d\t", remainder);
       num = num/10;
       sum = sum+remainder;
   }
   printf("\n 该数的所有数字之和为%d\n", sum);
}
```

程序运行情况如下：

```
请输入一个 4 位整数：4578↙
该数的各位数字,从个位开始依次为 8      7       5       4
该数的所有数字之和为 24
```

4.3　编写一个函数实现用弦截法求方程 $x^3-3x^2+3x-9=0$ 的近似根。主函数完成各系数值的输入及所求得的根值的输出。

解：算法思想（如图 1.4 所示）：

(1) 取两个不同点 x1, x2。如果 f(x1)和 f(x2)符号相反,则(x1,x2)区间内必有一个根；如果 f(x1)与 f(x2)同符号,则应改变 x1, x2,直到 f(x1)、f(x2)异号为止。注意 x1、x2 的值不应差太大,以保证(x1,x2)区间内只有一个根。

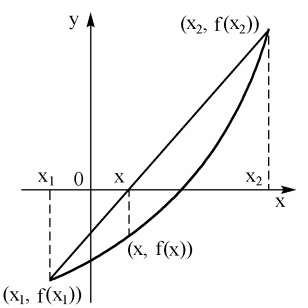

图 1.4　用弦截法求方程

(2) 连接(x1,f(x1))和(x2,f(x2))两点，此线（即弦）交 x 轴于 x。

(3) 若 f(x)与 f(x1)同符号，则根必在(x,x2)区间内，此时将 x 作为新的 x1。如果 f(x)与 f(x2)同符号，则表示根在(x1,x)区间内，将 x 作为新的 x2。

(4) 重复步骤(2)和(3)，直到 $|f(x)| < \varepsilon$为止，ε为一个很小的数，例如 10^{-6}。此时认为 $f(x) \approx 0$。

程序中定义的函数说明：

(1) 用函数 f(x)代表 x 的函数： $x^3 - 3x^2 + 3x - 9$ 。

(2) 用函数调用 point (x1,x2)来求(x1,f(x1))和(x2,f(x2))的连线与 x 轴的交点 x 的坐标。

(3) 用函数调用 root (x1,x2)来求(x1,x2)区间的那个实根。显然，执行 root 函数过程中要用到函数 point，而执行 point 函数过程中要用到 f 函数。

```
#include <stdio.h>
#include <math.h>
double f(double x);
double point(double x1, double x2);
double root(double x1, double x2);
void main()
{   double x1, x2, x, f1, f2;
    do
    {   printf("请输入 x1,x2:");
        scanf("%lf,%lf",&x1,&x2);
        f1=f(x1);
        f2=f(x2);
    }while(f1*f2>=0);       /* x1 和 x2 必须保证 f1*f2 <= 0，否则重新输入 */
    x = root(x1,x2);
    printf("方程的根是：%8.5f\n",x);
}
double f(double x)          /* 定义 f 函数，计算 f(x) = x3-3x2+3x-9 */
{   double y;
    y=((x-3.0)*x+3.0)* x-9.0;
    return (y);
}
double point(double x1, double x2)
{   double y;
    y=(x1*f(x2)-x2*f(x1))/(f(x2)-f(x1));
    return (y);
}
double root(double x1, double x2)
{   double x, y, y1;
    y1=f(x1);
    do
    {   x=point(x1, x2);
        y=f(x);
        if(y*y1>0)
            {   y1=y;
                x1=x;
            }
        else
            x2=x;
    } while (fabs(y)>=1e-6);
    return (x);
}
```

程序运行情况如下：

```
请输入 x1, x2:1,9↙
方程的根是： 3.00000
```

4.4　编写一个函数实现近似公式 $e=1+\frac{1}{1!}+\frac{1}{2!}+\frac{1}{3!}+L+\frac{1}{n!}$，计算自然对数的底 e 的近似值（n 从输入得到）。

解：

```
#include <stdio.h>
int Factorial(int n);
double Calce(int n);
void main()
{   int n;
    double e;
    do
    {   printf("请输入 n 的值：");
        scanf("%d", &n);
    }   while (n <= 0);          /* 保证 n 的值为正数 */
    e = Calce(n);
    printf("自然对数的底 e 的近似值为: %f\n", e);
}
double Calce(int n)              /* 计算 e 的值 */
{   int i;
    double r;
    double e=1.0;
    for (i=1;i<=n;i++)
    {   r=1.0/Factorial(i);
        e=e+r;
    }
    return (e);
}
int Factorial(int n)             /* 求阶乘函数，教材中的例 4.7 */
{   int product, i;
    product = 1;
    for (i=1;i<=n;i++)
        product*=i;
    return (product);
}
```

程序运行情况如下：

```
请输入 n 的值：10↙
自然对数的底 e 的近似值为: 2.718282
```

4.5　编写一个实现十进制数到十六进制数的转换函数，并试用主函数调用之。

解：十进制转换为十六进制的基本思想是：除 16 取余，直到除数为 0 为止。利用递归的思想编程实现，具体如下（该算法同样适用于十进制转换成二进制或八进制，请尝试写出其他两种函数）。

```
#include <stdio.h>
void DecimalToHex(int n);
void DisplayHex(int n);
void main()
{  int n;
```

```
    printf("请输入待转换的十进制数：");
    scanf("%d", &n);
    DecimalToHex(n);
}
void DecimalToHex(int n)
{  if (n<0)                    /* n 为负数时，在前面加负号 */
   {   printf("-");
       DecimalToHex(-n);
   }
   else if (n<16)              /* 对小于 16 的数，直接显示 */
           DisplayHex(n);
       else
       {   DecimalToHex(n/16);
           DisplayHex(n%16);
       }
}
void DisplayHex(int n)
{  if (n<10)                   /* 小于 10 的数，直接输出 */
       printf("%d", n);
   else                        /* 在 10 到 15 之间的数，输出其对应的字符 */
       switch(n)
       {   case 10: printf("A"); break;
           case 11: printf("B"); break;
           case 12: printf("C"); break;
           case 13: printf("D"); break;
           case 14: printf("E"); break;
           case 15: printf("F"); break;
       }
}
```

程序运行情况如下：

```
请输入待转换的十进制数：189↙
BD
```

4.6 编写一个函数判断正整数 n 是否是素数。

解：素数也称为质数，指的是一个正整数，除了能被它本身和 1 整除以外，不能被任何其他数整除。例如，2, 3, 5, 7 是质数，而 4, 6, 8, 9 则不是，后者称为合成数。判断某个数是否是素数，可以根据素数的性质，从最小的素数 2 开始，一直到比它小 1 的数为止，用这些数去除它，如果它能被整除，则它必定不是素数。如果进行深入的分析，会发现只需判断从 2 开始直到该数的开方的所有数即可。

```
#include <stdio.h>
#include <math.h>
int IsPrime(unsigned n);
void main()
{  int n;
   do
   {   printf("请输入一个正整数：");
       scanf("%d",&n);
   }while(n<=0);           /* 保证输入的是正数 */
   if (IsPrime(n))
       printf("%d 是素数!", n);
   else
```

```
        printf("%d 不是素数!", n);
    }
    /* 返回值为 1 表示该数为素数，为 0 表示该数不是素数 */
    int IsPrime(unsigned n)
    {  int i;
       int limit=(int)sqrt(n)+1;
       if (n==1)                /* 1 不是素数，tag 置为 0，返回 */
           return (0);
       else
           { /* 只要 2 到 K 之间存在一个数，能被 n 整除，则 n 就不是素数，即可结束循环 */
               for (i=2;i<=limit;i++)
               if (n%i==0)
                   return 0;
           }
       return (1);
    }
```

程序运行情况如下：

```
请输入一个正整数：11↙
11 是素数!
```

4.7　编写一个函数利用欧几里得算法（辗转相除法）求两个整数的最大公约数。

解：求 Greatest Common Divisor（GCD）也是一个经典的数学问题。欧几里得算法也称为辗转相除法，是由古代希腊最负盛名、最有影响的数学家欧几里得提出的。它的基本思想如下：

(1) 计算 x 除 y 的余数 r;

(2) 如果 r 是 0, 过程完成，结果就是 y;

(3) 如果 r 不是 0，设 x 等于原来的 y 值，y 等于 r，重复整个过程。

```
#include <stdio.h>
int GCD(int x, int y);
void main()
{   int x=49;
    int y=35;
    printf("%d 和 %d 的最大公约数是： %d\n", x, y, GCD(x, y));
}
int GCD(int x, int y)
{   int r;
    while(1)
    {   r=x%y;          /* 步骤 1，计算余数 r*/
        if(r==0)        /* 步骤 2，过程完成 */
            break;
        x=y;            /* 步骤 3, x 等于原来的 y 值，y 等于 r */
        y=r;
    }
    return (y);
}
```

程序运行情况如下：

```
49 和 35 的最大公约数是： 7
```

4.8　编写一个函数求两个整数的最小公倍数。

解：最小公倍数有一个简单的换算关系：最小公倍数=（a*b）/最大公约数。直接调用习题 4.7 中的 GCD 函数即可。

```
int lcm(int x, int y)
{   return (x*y/GCD(x,y));
}
```

4.9 编写一个函数判断某年是否是闰年。若是闰年，则函数返回 1，否则返回 0。

解：闰年的计算方法：公元纪年的年数可以被 4 整除，且不能被 100 整除即为闰年；或能被 400 整除的也为闰年。如 2000 年是闰年，而 1900 年则不是。

```
int IsLeapYear(int year)
{   return ((year%4==0)&&(year%100!=0) || (year%400==0));
}
```

return 语言中的表达式值为真，则函数 IsLeapYear 返回 1，否则返回 0。

4.10 用递归方法通过函数调用方式将一个整数转换成字符串，如整数 543 对应的字符串为“543”。

解：用递归的前提是，某个问题可以转换成相似的但规模更小的子问题，一直转换下去，直到某个问题足够小，不需要继续转换就能得出解，然后返回。

比如输入 543， 要转成“543”。问题等价于：(1) 把 54 转换为“54”，再在后面加上“3”；(2) 把 54 转换为“54”，等价于把 5 转换为“5”，再在后面加上“4”；(3) 把 5 转换为“5”，这个问题很容易解决，就是 5+'0'。

```
#include <stdio.h>
void IntToCharacters();
void main()
{   int n;
    printf("请输入要转换的整数：");
    scanf("%d",&n);
    IntToCharacters(n); /* 整数转换为字符，并显示 */
}
void IntToCharacters(int n)
{   char c;
    c= n%10+'0';
    if ((n/10)==0)
        putchar(c);
    else
    {   IntToCharacters(n/10);
        putchar(c);
    }
}
```

程序运行情况如下：

```
请输入要转换的整数：456↙
对应的字符串为 456
```

4.11 编写程序完成用递归方法求 n 阶勒让德多项式的值。递归公式为

$$H_n(x)=\begin{cases}1 & n=0\\ x & n=1\\ ((2n-1)xH_{n-1}(x)-(n-1)H_{n-1}(x))/n & n>1\end{cases}$$

解：

```
#include <stdio.h>
double H(int n, int x);
void main()
```

```
{   int n,x;
    printf("请输入多项式中 n 和 x 的值：\n");
    scanf("%d,%d",&n, &x);
    printf("当 n = %d，x = %d 时，多项式的值为%5.2f。\n", n, x, H(n, x));
}
double H(int n, int x)
{   if(n==0)
        return (1);
    else if(n==1)
        return (x);
    else
        return((2*n-1)*x*H(n-1,x)-(n-1)*H(n-2,x))/n;
}
```

4.12　有 n 个人围成一圈，顺序编号。从第 1 个人开始报数，从 1 报到 5，凡报到 5 的人退出圈子，问最后留下的人是原来的第几号？

解：该问题的实质是约瑟夫（Josephus）环问题。由于当某个人退出圆圈后，剩下的人仍然围成一个圆圈，报数的工作要从下一个人开始，继续从 1 开始报数。Josephus 问题是建立在历史学家 Josephus 一个报告的基础之上的，该报告讲述了他和 40 个士兵在公元 67 年被罗马军队包围期间签定的一个自杀协定，Josephus 建议每个人杀死他的邻居，他很聪明地使自己成为这些人中的最后一个，因此获得生还。

```
#include <stdio.h>
int Josephus(int n, int m);
void main()
{   int n,m;
    printf("请输入总人数 n 和报数最大值 m:\n");
    scanf("%d,%d",&n, &m);
    m=Josephus(n, m);
    printf("最后留下来的是第 %d 号\n", m);
}
int Josephus(int n, int m)
{   int i, r=0;
    for (i=2; i<=n;i++)
        r=(r+m)%i;
    return (r+1);
}
```

程序运行情况如下：

```
请输入总人数 n 和报数最大值 m:
17,5↙
最后留下来的是第 11 号
```

第 5 章　数组和字符串

5.1　请用冒泡排序法和选择排序法对数据“4，5，9，12，17，3，1”进行降序排序。

解：

```
#include <stdio.h>
#define  SIZE  7
void BubbleSort(int array[], int n);
void main()
{   int i;
```

```
    int array[SIZE] = {4,5,9,12,17,3,1};
    BubbleSort(array, SIZE);
    printf("应用冒泡排序后的结果是：\n");
    for (i=0;i<SIZE;i++)
        printf("%d\t",array[i]);
}
void BubbleSort(int array[], int n)/* array 为待排序数组，n 为元素个数 */
{   int i, j;
    int temp;
    for(i=0;i<n-1;i++)
        for(j=0;j<n-1-i;j++)
        {   if(array[j]<array[j+1])/*将原来的大于号改为小于号即按降序排序 */
            {   temp=array[j];
                array[j]=array[j+1];
                array[j+1]=temp;
            }
        }
}
```

程序运行情况如下：

```
应用冒泡排序后的结果是：
17    12    9     5     4     3     1
```

选择排序的实现如下：

```
void SelectSort(int array[], int n)/* array 为待排序数组，n 为元素个数 */
{   int i, j;
    int max_index;                  /* 存放值最大的元素的下标 */
    int temp;                       /* 元素交换时，存放中转数据 */
    for (i=0;i<n;i++)
    {   max_index = i;
        for (j=i+1;j<n;j++)         /* 查找值最大的元素的下标 */
        {   if(array[j]>array[max_index])
                max_index=j;
        }
        temp=array[i];              /* 交换元素 */
        array[i]=array[max_index];
        array[max_index]=temp;
    }
}
```

5.2 由于各评委在打分时存在某些主观因素，因此在计算平均值时通常去掉一个最高分，去掉一个最低分。写一个程序，读入 7 个评委所打的分数，去掉一个最低分，去掉一个最高分，求剩余 5 个分数的平均值。

解：在给出满足题目要求的程序后，将主教材例 5.4 整个程序进行了改写，具体内容如下：

```
#include <stdio.h>
#define NJudges   7                 /*  评委人数  */
#define NSingers  5
double CalcAverage(double scores[]);
void  main()
{   int i, j;
    double scores[NJudges + 1];     /* 存储每个评委评分，评委编号从 1 开始 */
    double average[NSingers + 1];   /* 存储歌手的平均分，歌手编号从 1 开始 */
    printf("------------请输入每个歌手的分数------------.\n");
```

```
    for (i=1;i<=NSingers;i++)
    {   printf("****第 %d 号歌手的得分情况****\n", i);
        for (j=1;j<=NJudges;j++)
        {   printf("第 %d 位评委打分: ", j);
            scanf("%lf", &scores[j]);
        }
        average[i]=CalcAverage(scores);
        printf("第 %d 号歌手的平均得分是: %.2f 分\n", i, average[i]);
    }
}
/* 找出评委的最高分和最低分，分别将其置为 0 */
double CalcAverage(double scores[])
{   int i;
    int min_index=1;
    int max_index=1;
    double average;
    double total=0;
    for(i=2; i<=NJudges ;i++)
    {   if(scores[min_index]>scores[i])
            min_index=i;            /* 保持最低分的下标 */
        if(scores[max_index]<scores[i])
            max_index=i;            /* 保持最高分的下标 */
    }
    scores[min_index]=0;            /* 最低分置为 0 */
    scores[max_index]=0;            /* 最高分置为 0 */
    for(i=1;i<=NJudges;i++)
        total+=scores[i];
    average=total/(NJudges-2);      /* 计算平均分，去掉最高分和最低分 */
    return(average);
}
```

5.3　从键盘输入 10 个整数并保存在数组中，然后计算它们的平均值，找出其中的最大数和最小数，并显示结果。

解：

```
#include <stdio.h>
#define SIZE 100
void GetIntArray(int Array[], int n);   /* 输入 n 个数，并保存在数组 Array */
int AverOfArray(int Array[],int n);     /* 计算 n 个数组元素的平均值 */
int MaxValueOfArray(int Array[],int n); /* 求含 n 个元素的数组的最大值 */
int MinValueOfArray(int Array[],int n); /* 求含 n 个元素的数组的最小值 */
void main()
{  int n=10;
   int Array[SIZE];
   GetIntArray(Array, n);
   printf("数组的平均值为%d\n", AverOfArray(Array, n));
   printf("数组元素的最大值为 %d， ",MaxValueOfArray(Array, n)) ;
   printf("最小值为 %d\n", MinValueOfArray(Array, n));
}
void GetIntArray(int Array[], int n)
{  int i;
   printf("请输入 %d 个数组元素的值：", n);
   for(i=0;i<n;i++)
       scanf("%d", &Array[i]);
}
int AverOfArray(int Array[],int n)
```

```
{  int i, average;
   int total = 0;
   for(i=0;i<n;i++)
       total+=Array[i];
   average=total/n;
   return(average);
}
int MaxValueOfArray(int Array[],int n)
{  int i;
   int max_index = 0;
   for (i=0;i<n;i++ )
       if (Array[max_index]<Array[i])
           max_index=i;
   return (Array[max_index]);
}
int MinValueOfArray(int Array[], int n)
{  int i;
   int min_index=0;
   for (i=0; i<n;i++ )
       if (Array[min_index]>Array[i])
           min_index=i;
   return (Array[min_index]);
}
```

程序运行情况如下：

```
请输入 10 个数组元素的值：4 5 9 12 17 3 1 0 -8 -55↙
数组的平均值为-1
数组元素的最大值为 17，最小值为 -55
```

5.4 有一整型二维数组 a[10][10]，按下列要求写出各题的 C 语言程序段。

(1) 按行输出所有的数组元素。

(2) 按列输出所有的数组元素。

(3) 输出主对角线上的所有元素。

(4) 输出副对角线上的所有元素。

(5) 输出上三角阵（包含主对角线元素）的所有元素。

(6) 输出上三角阵（包含副对角线元素）的所有元素。

(7) 输出下三角阵（包含主对角线元素）的所有元素。

(8) 输出下三角阵（包含副对角线元素）的所有元素。

解：(1) 按行输出所有的数组元素。

```
for(i=0;i<10;i++)
{   for (j=0;j<10;j++)
        printf("%5d",a[i][j] );
    printf("\n");
}
```

(2) 按列输出所有的数组元素。

```
for(i=0;i<10;i++)
{   for (j=0;j<10;j++)
        printf("%5d",a[j][i] );
    printf("\n");
}
```

(3) 输出主对角线上的所有元素。

```
for(i=0;i<10;i++)
{   for(j=0;j<10;j++)
    {   if(i==j)
            printf("%5d",a[i][j] );
        else
            printf("    ");
    }
    printf("\n");
}
```

(4) 输出副对角线上的所有元素。

```
for(i=0;i<10;i++)
{   for (j=0;j<10;j++)
    {   if((i+j)==9)
            printf("%5d",a[i][j] );
        else
            printf("    ");
    }
    printf("\n");
}
```

(5) 输出上三角阵（包含主对角线元素）的所有元素。

```
for(i=0;i<10;i++)
{   for(j=0;j<10;j++)
    {   if(i<=j)
            printf("%5d",a[i][j] );
        else
            printf("    ");
    }
    printf("\n");
}
```

(6) 输出上三角阵（包含副对角线元素）的所有元素。

```
for(i=0;i<10;i++)
{   for (j=0;j<10;j++)
    {   if(j<10-i)
            printf("%5d",a[i][j] );
        else
            printf("    ");
    }
    printf("\n");
}
```

(7) 输出下三角阵（包含主对角线元素）的所有元素。

```
for(i=0;i<10;i++)
{   for(j=0;j<10;j++)
    {   if(i>=j)
            printf("%5d",a[i][j] );
        else
            printf("    ");
    }
    printf("\n");
}
```

(8) 输出下三角阵（包含副对角线元素）的所有元素。

```
for(i=0;i<10;i++)
{   for(j=0;j<10;j++)
    {   if(j>=9-i)
            printf("%5d",a[i][j] );
        else
            printf("    ");
        }
        printf("\n");
    }
```

5.5 按如下图案打印杨辉三角形的前 10 行。（杨辉三角形是由二项式定理系数表组成的图形, 其特点是两个腰上的数都为 1，其他位置上的每一个数是它上一行相邻的两个整数之和。）

```
                1
              1   1
            1   2   1
          1   3   3   1
        1   4   6   4   1
      1   5  10  10  5   1
          ......
```

解：

```
#include <stdio.h>
void main()
{   int a[10][10]={0};              /* 定义二维数组，元素初值全为 0 */
    int i,j;
    for (i=0;i<10;i++)
        a[i][0]=1;                  /* 第 0 列全为 1 */
    for (i=1;i<10;i++)
        for (j=1;j<=i;j++)
            a[i][j]=a[i-1][j-1]+a[i-1][j];
                                    /* 每一个数是它上一行相邻的两个整数之和 */
    for(i=0;i<10;i++)
    {   for(j=0;j<10-i;j++)
            printf("   ");          /* 第 i 行数据前显示 3*(10-i)个空格 */
        for(j=0;j<=i;j++)
            printf("%6d",a[i][j]); /* 第 i 行输出 i 个数据 */
        printf("\n");               /* 输出第 i 行数据后回车 */
    }
}
```

程序运行情况如下：

```
                    1
                  1   1
                1   2   1
              1   3   3   1
            1   4   6   4   1
          1   5  10  10   5   1
        1   6  15  20  15   6   1
      1   7  21  35  35  21   7   1
    1   8  28  56  70  56  28   8   1
  1   9  36  84 126 126  84  36   9   1
```

5.6 编写一个函数,求一个二维矩阵的转置矩阵，即将原矩阵行列互换的结果。

解：

```
#include <stdio.h>
```

```
#define ROW  3
#define COL  3
void main()
{   int Matrix_A[ROW][COL];
    int Matrix_B[COL][ROW];
    int i, j;
    printf("请输入 %d 行 %d 列二维矩阵的初始值：\n", ROW, COL );
    for(i=0;i<ROW;i++)
        for(j=0;j<COL;j++)
            scanf("%d", &Matrix_A[i][j]);
    printf("转置前的矩阵为\n");
    for(i=0;i<ROW;i++)
    {   for(j=0;j<COL;j++)
            printf("%5d", Matrix_A[i][j]);
        printf("\n");
    }
    for(i=0;i<ROW;i++)
        for(j=0;j<COL;j++)
            Matrix_B[j][i] = Matrix_A[i][j];    /* 矩阵转置 */
    printf("转置后的矩阵为\n");
    for(i=0;i<ROW;i++)
    {   for(j=0;j<COL;j++)
            printf("%5d", Matrix_B[i][j]);
        printf("\n");
    }
}
```

程序运行情况如下：

```
请输入 2 行 3 列二维矩阵的初始值：
1↙
2↙
3↙
4↙
5↙
6↙
转置前的矩阵为
    1    2    3
    4    5    6
转置后的矩阵为
    1    4
    2    5
    3    6
```

5.7　在下面的 5×5 魔方矩阵中，每一行、每一列、每一对角线上的元素之和都是相等的，试编写程序将这些魔方矩阵中的元素读到一个二维整型数组中，然后，检验其是否为魔方矩阵，并将其按如下格式显示到屏幕上。

17	24	1	8	15
23	5	7	14	16
4	6	13	20	22
10	12	19	21	3
11	18	25	2	9

解：
```
#include <stdio.h>
#define N  16
void main()
{   int array[N][N],i,j,k,n;
    do
{   printf("请输入魔方矩阵的阶数 n，要求 n 在 0 到 16 之间且 n 是奇数:");
    scanf("%d",&n);
}while(n<0||n>16||(n%2)==0);          /* 若 n 不合要求，则重新输入 */
for(i=1;i<=n;i++)                     /* 魔方矩阵初值皆为 0 */
    for(j=1;j<=n;j++)
        array[i][j]=0;
    /*建立魔方矩阵*/
    j=(n/2)+1;
    array[1][j]=1;
    for(k=2;k<=n * n; k++)
    {   i=i-1; j=j+ 1;
        if((i<1)&&(j>n))
        {   i=i+2; j=j-1; }
        else
        {   if(i<1)   i=n;
            if(j>n)   j=1;
        }
        if(array[i][j]==0)
            array[i][j]=k;
        else
        {   i=i+2; j=j-1; array[i][j]=k;  }
    }
    /*输出 n 阶魔方矩阵*/
    printf("%d 阶魔方矩阵为\n", n);
    for(i=1;i<=n;i++)
    {   for(j=1;j<=n;j++)
            printf("%-5d",array[i][j]);
        printf("\n");
    }
}
```

程序运行情况如下：

```
请输入魔方矩阵的阶数 n，要求 n 在 0 到 16 之间且 n 是奇数。
5↙
5 阶魔方矩阵为
17    24    1    8    15
23    5     7    14   16
4     6     13   20   22
10    12    19   21   3
11    18    25   2    9
```

5.8　编写一个函数实现 strcmp 的功能，将两个字符串 s 和 t 进行比较，然后将两个字符串中第一个不相同字符的 ASCII 码值之差作为函数值返回。

解：
```
int MyStrCmp(char s[], char t[])
{   int i;
    for(i=0;s[i]==t[i];i++)
        if(s[i]=='\0') return (0);
    return(s[i]-t[i]);
}
```

5.9　编写一个函数，要求函数的功能为删除字符串 s 中所出现的与变量 c 相同的字符，假设函数原型为 void Sdeletec(char s[],char c)。

解：
```
void Sdeletec(char s[], char c)
{   int i, j;
    for(i=0;s[i]!='\0';i++)
        if(s[i]==c)
        {   for(j=i;s[j]!='\0';j++)
                s[j]=s[j+1];
            s[j-1]=s[j];            /* 移动串结束标志符*/
        }
}
```

5.10　编写一个函数实现字符串连接函数 strcat 的功能，将字符串 t 连接到字符串 s 的尾部。

解：
```
void StrCat(char s[], char t[])
{  int i, j;
   for (i=0;s[i]!='\0';i++)        /* 移动数组下标，使其指向串结束标志符*/
       ;
   for (j=0;t[j]!='\0';j++,i++)
       s[i]=t[j];
   s[i]=t[j];                      /* 复制 t 串的串结束标志符*/
}
```

5.11　将一行字符中的每个单词的第一个字母改成大写。

解：
```
#include <stdio.h>
#include <string.h>
#define LENGTH 100
void MyString(char s[]);
void main()
{  char str[LENGTH];
   printf("请输入字符串：");
   gets(str);
   MyString(str);
   printf("每个单词首字母都改成大写后的字符串为\n");
   puts(str);
}
void MyString(char s[])
{  int i;
   for (i=0;s[i]!='\0';i++)
       if (s[i]==' ' && s[i+1]!='\0')
           if (s[i+1]>='a' && s[i+1]<='z')
               s[i+1]-='a'-'A';
}
```

程序运行情况如下：

```
请输入字符串：No one can avoid being influnced by advertisement↙
每个单词首字母都改成大写后的字符串为
No One Can Avoid Being Influnced By Advertisement
```

第 6 章　指　　针

本章所有编程题要求用指针处理。

6.1　指针变量和变量的指针有何异同？

解：指针变量：专门存放地址的一类特殊变量；变量的指针：指变量在内存中的地址。

6.2 指针运算的实质是什么？对指针可以实行哪些运算？

解：指针运算实质是地址运算。

指针运算有：

(1) 两个指针变量相减；

(2) 一个指针变量加上（或减去）一个整数；

(3) 指针变量的赋值运算；

(4) 指针赋空值；

(5) 两个指针变量的关系运算（可比较大小）。

6.3 设有下面程序段，哪些语句是非法的？

(1)
```
int i,j,*p,*q;
i=2;
P=&i;
j=*q++**p;
```

(2)
```
int pa,pb,a[4],b[4]={1,2,3};
pa=a;
*pb=b;
pa[0]=b[0];
*++pa=4;
a[2]=b[2];
*(pa+3)=5;
```

(3)
```
swap(int *px, int *py)
{   int temp;
    temp=px;
    px=py;
    py=temp;
}
```

解：(1) 第 3 行 p 大写错误，最后一行的指针 q 未指向任何变量不能做*q 的运算。

(2) 第 1 行 pa，pb 没有定义为指针，第 3 行*pb＝b 错误，应为 pb＝b。

(3) 无法起到交换数据的作用。

6.4 编写一个程序，输入 3 个字符串，将它们按由小到大的顺序输出。

解：
```
#include <stdio.h>
#include <string.h>
void main()
{   char a[30],b[30],c[30];
    char *pa,*pb,*pc;
    scanf("%s%s%s",a,b,c);          /* 输入三个字符串*/
    if (strcmp(a,b)>0)
        if (strcmp(b,c)>0)
            {   pa=a;pb=b;pc=c;     /* 若字符串关系为 a>b>c，则令 pa 指向 a 字
                                       符串，pb 指向 b 字符串，pc 指向 c 字符串*/
            }
        else if (strcmp(c,a)>0)
                {   pa=c;pb=a;pc=b; /* 若字符串关系为 c>a>b，则令 pa 指向 c 字
                                       符串，pb 指向 a 字符串，pc 指向 b 字符串*/
                }
            else
```

```
                {   pa=a;pb=c;pc=b; /* 若字符串关系为 a>c>b，则令 pa 指向 a 字
                                       符串,pb 指向 c 字符串,pc 指向 b 字符串 */
                }
        else
            if (strcmp(b,c)<0)
            {   pa=c;pb=b;pc=a;         /* 若字符串关系为 a<b<c，则令 pa 指向 c 字
                                           符串,pb 指向 b 字符串,pc 指向 a 字符串 */
            }
            else if(strcmp(a,c)>0)
                { pa=b;pb=a;pc=c;       /* 若字符串关系为 c<a<b，则令 pa 指向 b 字
                                           符串,pb 指向 a 字符串,pc 指向 c 字符串 */
                }
                else
                {   pa=b;pb=c;pc=a;     /* 若字符串关系为 a<c<b，则令 pa 指向 b 字
                                           符串,pb 指向 c 字符串,pc 指向 a 字符串 */
                }
        printf("%s %s %s",pc, pb, pa);  /* 把字符串按由小到大的顺序输出 */
    }
```

程序运行情况如下：

```
beijing shanghai liuzhou↙
beijing liuzhou shanghai
```

6.5　输入 3 个整数，按由小到大的顺序输出。

解：

```
#include <stdio.h>
void main()
{   int a,b,c;
    int *pa,*pb,*pc;
    scanf("%d%d%d",&a,&b,&c);
    if (a>b)
        if (b>c)
        {   pa=&a;pb=&b;pc=&c; }      /*a>b>c 时 pa 指向 a，pb 指向 b，pc 指向 c*/
        else if (c>a)
            {   pa=&c;pb=&a;pc=&b;}  /*c>a>b 时 pa 指向 c，pb 指向 a，pc 指向 b*/
        else
            {   pa=&a;pb=&c;pc=&b;}  /*a>c>b 时 pa 指向 a，pb 指向 c，pc 指向 b*/
    else if (b<c)
        {    pa=&c;pb=&b;pc=&a; }     /*a<b<c 时 pa 指向 c，pb 指向 b，pc 指向 a*/
    else if(a>c)
        {   pa=&b;pb=&a;pc=&c;}       /*c<a<b 时 pa 指向 b，pb 指向 a，pc 指向 c*/
    else
        {   pa=&b;pb=&c;pc=&a;}       /*a<c<b 时 pa 指向 b，pb 指向 c，pc 指向 a*/
    printf("%d %d %d",*pc, *pb, *pa);
}
```

程序运行情况如下：

```
4 12 2↙
2 4  12
```

6.6　编写函数，对传送过来的三个数选出最大和最小数，并通过形参传回调用函数。

解：

```
#include <stdio.h>
#include <string.h>
void func(int a, int b, int c, int *max, int *min)
{   int *pa,*pb,*pc;
```

```
    if(a>b)
        if(b>c)
                {   pa=&a;pb=&b;pc=&c;  }
        else if (c>a)
                {   pa=&c;pb=&a;pc=&b;  }
        else
                {   pa=&a;pb=&c;pc=&b;  }
    else if (b<c)
        {   pa=&c;pb=&b;pc=&a;   }
    else if(a>c)
        {   pa=&b;pb=&a;pc=&c;  }
    else
        {   pa=&b;pb=&c;pc=&a;  }
    *max=*pa;               /* 通过形参 max 传回最大数 */
    *min=*pc;               /* 通过形参 min 传回最小数 */
}
void main()
{   int a=1,b=2,c=3;
    int max, min;
    func(a,b,c,&max, &min); /* 实参 max，min 为传地址方式 */
    printf("max=%d min=%d",max, min);
}
```

程序运行情况如下：

```
 max=3 min=1
```

6.7 求一个 3×3 二维数组主对角线元素之和。

解：

```
#include <stdio.h>
void main()
{   int i,*p,a[3][3],sum=0;
    printf("请输入一个 3×3 的矩阵:\n");
    for(i=0;i<3;i++)
    for(p=a[i];p<a[i]+3;p++)
        scanf("%d",p);
    for(i=0;i<3;i++)            /* 主对角线元素的行号与列号相等 */
    {   p=a[i]+i;
        sum=sum+*p;
    }
    printf("3×3 的矩阵是:\n");
    for(i=0;i<3;i++)
    {   for(p=a[i];p<a[i]+3;p++)   /* 打印第 i 行*/
        printf("%4d",*p);
      printf("\n");
    }
  printf("主对角线元素之和=%d\n",sum);
}
```

程序运行情况如下：

```
请输入一个 3×3 的矩阵:
1 2 3↙
4 5 6↙
7 8 9↙
3×3 的矩阵是:
```

```
    1   2   3
    4   5   6
    7   8   9
主对角线元素之和=15
```

6.8　将一个 3×3 二维数组转置。

解：

```
#include <stdio.h>
#define N  3
void main()
{   int a[N][N]={3,5,6,5,8,9,3,1,2};
        int i,j,temp;
        int *p[N];      /* 定义一个指针数组 */
    for(i=0;i<N;i++)
        p[i]=a[i];      /*使指针数组 p 指向数组 a，它们之间的关系如图 1.5 所示 */
    printf("原始的矩阵:\n");
    for(i=0;i<N;i++)
    {   for(j=0;j<N;j++)
            printf("%3d",p[i][j]);
        printf("\n");
    }
    for(i=0;i<N;i++)
        for(j=i+1;j<N;j++)
        {   temp=p[i][j];p[i][j]=p[j][i];p[j][i]=temp;}/*行列元素互换 */
    printf("转置后的矩阵:\n");
    for(i=0;i<N;i++)
    {   for(j=0;j<N;j++)
            printf("%3d",p[i][j]);          /* 输出转置矩阵 */
        printf("\n");
    }
}
```

程序运行情况如下：

```
原始的矩阵:
  3  5  6
  5  8  9
  3  1  2
转置后的矩阵:
  3  5  3
  5  8  1
  6  9  2
```

6.9　有 n 个整数，使前面各数顺序向后移动 m 个位置，最后 m 个数变成最前面 m 个数，如图 1.6 所示。编写一个函数实现以上功能，在主函数中输入 n 个整数和输出调整后的 n 个数。

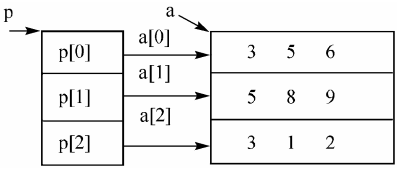

图 1.5　习题 6.8 数组、指针关系的示意图

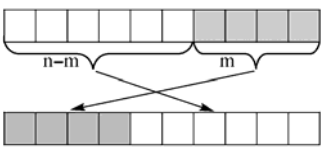

图 1.6　习题 6.9 的示意图

解：先移动最后一个元素到变量 k 中，然后把前面的元素依次移动一个位置，再把 k 值赋给最前面的元素，其余 m–1 个数按同样的方法处理。

```
#include <stdio.h>
void move(int *a,int n, int m);
void main()
{   int a[10],i,m;
    printf("请输入 10 个数：\n");
    for(i=0;i<10;i++)
        scanf("%d",&a[i]);
    printf("\n 请输入移动位置个数：");
    scanf("%d",&m);
    move(a,10,m);
    printf("移动后的 10 个数：\n");
    for(i=0;i<10;i++)
        printf("%4d",a[i]);
}
void move(int *a,int n, int m)
{   int i,*p,*q,k;
    for(i=0;i<m;i++)
        {   q=a+n-1;          /* q 指向最后一个元素 */
            p=q-1;            /* p 指向 q 的前一个元素 */
            k=*q;             /* 暂存最后一个数 */
            while(p>=a)       /* 前面的 n-1 个数依次后移一个位置 */
            {   *q=*p;
                p--;
                q--;
            }
            a[0]=k;           /* 原最后元素移到最前面 */
        }
}
```

程序运行情况如下：

```
请输入 10 个数：
1 2 3 4 5 6 7 8 9 10
请输入移动位置个数：4
移动后的 10 个数：
  7   8   9  10   1   2   3   4   5   6
```

6.10　用指针参数编写一个字符串连接的函数 radd(char *s,char *t,int f)，其中 f 为标志变量，当 f = 0 时，将 s 指向的字符串连接到 t 指向的字符串的后面；当 f = 1 时，将 t 指向的字符串连接到 s 指向的字符串的后面。写出调用该函数的完整程序。

解：

```
#include <stdio.h>
#include <string.h>
void radd(char *s,char *t,int f)
{   if(f==0)
        strcat(t,s);    /*调用字符串连接库函数 strcat()*/
    if (f==1)
        strcat(s,t);
}
void main()
{   char s[100]="test1";
    char t[100]="test2";
```

```
    int choice;
    scanf("%d",&choice);
    if (choice==0)
        {   radd(s,t,0);
            printf("t+s=>t : %s\n",t);
    }
else if(choice==1)
     {  radd(s,t,1);
        printf("s+t=>s : %s\n",s);
     }
}
```

程序运行情况如下：

```
0↙
t+s=>t : test2test1
```

再运行一次：

```
1↙
s+t=>s: test1test2
```

6.11　从一个给定的字符串中找出某一子字符串的位置（从 1 开始）。例如，子串"efg"在字符串"abcdefghijk"中位置为 5。若字符串中没有指定的子串，则标记为 0。

解：

```
#include <stdio.h>
void main()
{   int b=0,p;              /* b 等于 0 表示还未找到子串 */
    char str[80],substr[80],*s,*r,*t;
    printf("请输入字符串:\n");
    gets(str);
    printf("请输入子字符串:\n");
    gets(substr);
    s=str;
    t=substr;
    printf("子串%s 在字符串%s 中位置为",substr,str);
    for(s=str,p=1;*s!='\0';s++,p++)
    {   for(t=substr,r=s;*t!='\0'&&*t==*r;t++,r++);
        if(*t=='\0')
    {   b=1;                /* b 等于 1 表示已找到一个子串 */
        printf("%d  ",p);
    }
    }
    if(b==0)
    printf("0");
    printf("\n");
}
```

程序运行情况如下：

```
请输入字符串:
abcdefghijk↙
请输入子字符串:
efg↙
子串 efg 在字符串 abcdefghijk 中位置为 5
```

6.12　求出 5 个字符串中最长的字符串。

解：方法 1：

```
#include <stdio.h>
#include <string.h>
void main()
{   int i,k,len[5];     /* k 记最长字符串的位置，len[i]记第 i 个字符串的长度 */
    char str[5][80],(*c)[80];
    c=str;
    printf("请输入 5 个字符串:\n");
    for(i=0;i<5;i++)
    {   gets(*(c+i));
        len[i]=strlen(*(c+i));
    }
    k=0;
    for(i=1;i<5;i++)
        if(len[i]>len[k])
            k=i;
    printf("最长的字符串是：%s\n",*(c+k));
}
```

程序运行情况如下：

```
请输入 5 个字符串:
Pascal↙
Basic↙
Java↙
Fortran↙
C↙
最长的字符串是：fortran
```

方法 2：

```
#include <stdio.h>
#include <string.h>
void main()
{   char *name[]={"Pascal","Basic","Java","Fortran","C"};
    char *ptr;
    int i,length,max;            /*max 为 5 个串中最大的长度，length 为串长 */
    ptr=name;
    max=strlen(name[0]);        /* 第 0 个字符串的长度送给变量 max */
    for(i=1;i<5;i++)
    if(max<(length=strlen(name[i])))
    {   max=length;
        ptr=name[i];
    }
    printf("最长的字符串是：%s\n",ptr);   /* 输出最大的字符串 */
}
```

程序运行情况如下：

```
最长的字符串是：Fortran
```

6.13　找出一个字符串中最大的字符并把它放在最前面，其他字符往后顺序存放。如字符串"student"处理后成为"ustdent"。

解：

```
#include <stdio.h>
void main()
{   char *p,*q, str[80], ch;
    int i;
```

```
    printf("请输入一个字符串：");
    gets(str);
    p=str;
    q=p;
    i=0;
    printf("原字符串为%s",str);
    while(*q!=NULL)      /* 逐个比较，寻找最大的字符 */
    {   if(*(p+i)< *q)
        i=q-p;           /* 把当前比较所得的最大字符的位置记在 i 变量中 */
        q++;             /* 取下一个字符 */
    }
    ch=*(p+i);           /* ch 暂存最大字符 */
    for(;i>0;i--)        /* 其他字符往后顺序存放 */
    *(p+i)= *(p+i-1);
    *p=ch;               /* 把最大字符放在字符串的最前面 */
    printf("\n 处理后的字符串为%s", p);
}
```

程序运行情况如下：

```
请输入一个字符串：student↙
原字符串为 student
处理后的字符串为 ustdent
```

6.14　不使用库函数 strcmp 函数实现对两个字符串 s1 和 s2 进行比较。若 s1 和 s2 相等，输出 0；若它们不相等，则指出其第一个不同字符的 ASCII 码的差值：如果 s1>s2，则差值为正；如果 s1<s2，则差值为负。

解：

```
#include <stdio.h>
void  main()
{   char s1[80],s2[80], *p1, *p2;
    int n,i;
    printf("输入第一个字符串：");
    gets(s1);
    printf("输入第二个字符串：");
    gets(s2);
    p1=s1;p2=s2;         /* p1、p2 分别指向字符串 s1、s2 */
    while(*p1==*p2)      /* 逐个字符比较，若相等则继续 */
      { if(*p1=='\0')    /* 若相等且比较到串尾则退出循环 */
            break;
        p1++;            /* 若相等且不到串尾则分别移动指针 */
        p2++;
      }
    n=*p1-*p2;           /* 当字符不等或为'\0'结束循环后，求出最后比较字符之差 */
    printf("两个字符串比较的结果是：%d\n",n);
}
```

程序运行情况如下：

```
输入第一个字符串：java↙
输入第二个字符串：basic↙
两个字符串比较的结果是：8
```

6.15　输入 10 个数，找出其中的最大值和最小值。

解：

```
#include <stdio.h>
void func(int a[],int *max, int *min)
```

```
{  int i;
   *max=*a;                  /* 假设最大值是 a 的第一个元素 */
   *min=*a;                  /* 假设最小值是 a 的第一个元素 */
   for(i=1;i<10;i++)         /* 通过循环逐一比较找出最大最小值 */
   if(*(a+i)> *max)
       *max=*(a+i);
   else if(*(a+i)< *min)
           *min=*(a+i);
}
void  main()
{  int a[10],i,max,min;
   printf("请输入 10 个整数：");
   for(i=0;i<10;i++)
       scanf("%d",&a[i]);
   func(a,&max, &min);      /* 实参 max，min 为传地址方式 */
   printf("max=%d min=%d",max, min);
}
```

程序运行情况如下：

```
请输入 10 个整数：1 2 3 4 60 5 7 5 -10 2↙
ax=60 min=-10
```

6.16 编写程序，将两个字符串合并到一个字符串中。

解：方法一：

```
#include <stdio.h>
void  main()
{   char str1[100],str2[100], *p1, *p2;
    int i;
    printf("输入第一个字符串：");
    gets(str1);
    printf("输入第二个字符串：");
    gets(str2);
    p1=str1;p2=str2;            /* p1、p2 分别指向字符串 str1、str2 */
    while(*p1!='\0')
        p1++;
    while((*p1++=*p2++)!='\0');
    *p1='\0';
    printf("合并后的字符串是：%s\n",str1);
}
```

方法二：利用返回指针值的函数实现两个字符串的合并。

```
#include <stdio.h>
char *link(char *p1,char *p2);
void  main()
{   char str1[100],str2[100], *p1, *p2;
    int i;
    printf("输入第一个字符串：");
    gets(str1);
    printf("输入第二个字符串：");
    gets(str2);
    printf("合并后的字符串是：%s\n", link(str1,str2));
}
char *link(char *p1,char *p2)
```

```
{   char *p;
    p=p1;
    while(*p1!='\0')
        p1++;
    while(*p2!='\0')
    {   *p1=*p2;
        p1++;p2++;
    }
    *p1='\0';
    return (p);
}
```

程序运行情况如下：

```
输入第一个字符串：abcde↙
输入第二个字符串：fghijklmn↙
合并后的字符串是：abcdefghijklmn
```

6.17　利用指向函数的指针实现求两个整数的差值。

解：

```
#include <stdio.h>
int f(int x,int y);
void  main()
{  int a,b,c,( *g)();                  /* 定义指向函数的指针 g */
   printf("请输入两个整数：\n");
   scanf("%d%d",&a,&b);
   g=f;                                /* 使指针指向 f 函数 */
   c=(*g)(a,b);                        /* 通过函数指针调用 f 函数 */
   printf("%d-%d=%d\n",b,a,c);
}
int f(int x,int y)
{  return (y-x);
}
```

程序运行情况如下：

```
请输入两个整数：
45 6↙
6-45=-39
```

6.18　编写程序，利用指向函数的指针实现求 1 到 n 的和与阶乘。

解：

```
#include <stdio.h>
int sum(int n);
int fac(int n);
int  choice(int(*f)(int n),int n); /* 函数声明 */
void  main()
{  int n;
   printf("请输入一个整数：");
   scanf("%d",&n);
   printf("从 1 到%d 的和是：%d\n ",n,choice(sum,n));
   printf("从 1 到%d 的阶乘是：%d\n",n,choice(fac,n));
}
int  choice(int (*f)(int n),int n) /* 第一个参数为指向函数的指针 */
{  return((*f)(n));
}
int sum(int n)                          /* 求 1 到 n 的和 */
{  int i,s;
```

```
    s=0;
    for(i=1;i<=n;i++)
      s+=i;
    return (s);
}
int fac(int n)                          /* 求 1 到 n 的阶乘 */
{  int i,s;
    s=1;
    for(i=1;i<=n;i++)
      s*=i;
    return (s);
}
```

程序运行情况如下：

```
请输入一个整数：5↙
从 1 到 5 的和是：15
从 1 到 5 的阶乘是：120
```

6.19　编写一个函数 new，对 n 个字符开辟连续的存储空间，此函数应返回一个指针（地址），指向字符串开始的空间。new(n)表示分配 n 个字节的内存空间，如图 1.7 所示。

解：new 函数是分配 n 个连续字符的存储区，为此，应先开辟一个足够大的连续存储区，设置字符数组 newbuf[1000]，new 函数将在此上进行操作。newp 是指向可用存储区的起始地址的指针。每当请求 new 开出 n 个字符的存储区时，要先检查一下 newbuf 是否还有足够的可用空间。若有，则返回指针 newp 的当前值，而后 newp+n 指向下一次可用空间的开始地址；若存储不够分配，则返回 NULL。可编写出以下 new 函数。

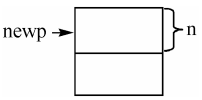

图 1.7　习题 6.19 的示意图

```
#define NULL 0
#define NEWSIZE 1000
char newbuf[NEWSIZE];
char *newp=newbuf;
char  *new(int n)
{   if (newp+n<=newbuf+NEWSIZE)
    {   newp+=n;
        return(newp-n);
    }
    else
        return(NULL);
}
```

6.20　编写一个函数 free，将第 6.19 题用 new 函数申请的空间释放。free(p)表示将 p（地址）指向的单元以后的内存段释放。

解：free 的作用是使 newp 的值改为 p。

```
#define NULL 0
#define NEWSIZE 1000
char newbuf[NEWSIZE];
char *newp=newbuf;
free(char *p)
{   if((p>=newbuf) && (p<newbuf+NEWSIZE))       newp=p;
}
```

6.21　利用 malloc 函数开辟动态存储单元，存放输入的 10 个整数，然后按升序输出这 10 个数。

解：

```
#include <stdio.h>
#include <stdlib.h>
#define N  10
struct int_link                          /*定义一个链接结构，用来存放输入的整数 */
{   int n;
    struct int_link  *next;
};
void main()
{   struct int_link *head, *p1, *p2, *p0;
    int i;
    head=malloc(sizeof(struct int_link));   /* 生成一个头指针 */
    head->next=NULL;
    printf("请输入 10 个数:\n");
    scanf("%d",&head->n);                    /* 输入第一个数 */
    for(i=1;i<N;i++)           /* 输入其他数，在输入的时候使其按升序排列 */
    {   p0=malloc(sizeof(struct int_link));
        p0->next=NULL;
        scanf("%d",&p0->n);
        p1=head;
        /* 为刚输入的数寻找一个合适的位置 */
        while((p0->n>p1->n)&&(p1->next!=NULL))
        {   p2=p1;
            p1=p1->next;
        }
        if(p0->n<p1->n)                      /* 将该数插入 */
          { if(head==p1)  head=p0;
            else p2->next=p0;
            p0->next=p1;
          }
        else
          { p1->next=p0;p0->next=NULL;}
    }
    for(p1=head;p1!=NULL;)                   /* 将排好序的链表输出 */
    {   printf("%d",p1->n);
        p1=p1->next;
    }
}
```

程序运行情况如下：

```
请输入 10 个数:
12 32 5 64 8 95 62 23 45 6↙
5 6 8 12 23 32 45 62 64 95
```

第 7 章　模块化程序设计

7.1　分析下面的函数定义和变量声明：

```
int a;
void f(int b)
{   int c; ..... }
void g(void)
```

```
{   int d;
    {   int e; ..... }
}
void main()
{   int f; ..... }
```

请列出在下面作用域内，有效的变量名（包括形参名）：(1) f 函数，(2) g 函数，(3) 声明 e 的程序块，(4) main 函数。

解：(1) f 函数中有效的变量是：a, b, c;

(2) g 函数中有效的变量是：a, d, e;

(3) 在声明 e 的程序块中有效的变量是：a, d, e;

(4) main 函数中有效的变量是：a, f。

7.2 若有宏定义 #define max(a,b) ((a)>(b)?(a):(b))

写出表达式 max(x,max(y,z))扩展后的形式。

解：表达式 max(x,max(y,z))扩展后的形式为

((x)>(((y)>(z)?(y):(z)))?(x):((y)>(z)?(y):(z)))

7.3 编写程序实现：主函数接受从键盘输入的年、月、日数据，计算该日是该年的第几日。

解：以 3 月 5 日为例，应该先把前两个月的日数加起来，然后再加上 5 日，所得出的和即是本年的第几日。需要考虑一个特殊情况，即闰年时 2 月份为 29 日，平年只有 28 日。

```
#include <stdio.h>
int IsLeapYear(int year);
void main()
{   int year, month, day, dayth, leap, i;
    int month_table[2][13]={{0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31,
            30, 31}, {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}};
    printf("请输入年，月，日（它们之间以逗号隔开）:");
    scanf("%d,%d,%d", &year, &month, &day);
    leap=IsLeapYear(year);
    dayth=day;
    for(i=0;i<month;i++)
        dayth += month_table[leap][i];
    printf("%d 年%d 月%d 日是该年的第 %d 日\n", year, month, day, dayth);
}
int IsLeapYear(int year)
{   return ((year%4==0) && (year & 100!=0) || (year%400==0));
}
```

程序运行情况如下：

```
请输入年，月，日（它们之间以逗号隔开）:2008,12,1↙
2008 年 12 月 1 日是该年的第 335 日
```

7.4 修改主教材例 7.8 中的程序，使得年份只能以两位数的形式输入（假设输入的年份都是 20 世纪的）。如果用户输入 95，则程序将产生 1995 年的年历。

解：

```
int GetYearFromUser(void)
{   int year;
    while(1)
    {   printf("请输入要显示的年份：");
```

```
            scanf("%d", &year);
            if(year>=0 && year<=99)
                return(1900+year);
            printf("请输入 1900 年以后的年份!\n");
        }
    }
```

7.5　修改主教材例 7.8 中的 FirstDayOfMonth 函数，使之能用于产生 1990 年前的年历，并思考程序的其他部分是否需要进行相应的修改。

解： 修改后的 FirstDayOfMonth 函数如下：

```
int FirstDayOfMonth(int month, int year)    /* 计算某年某月的第一天是星期几 */
{  int weekday, i;                          /* weekday 表示一周的某一天 */
   weekday = Monday;                        /* 从星期一开始计数 */
   if (year >= 1900)
       for (i = 1900; i < year; i++)
       {   weekday = (weekday + 365);
           if (IsLeapYear(i)) weekday = (weekday + 1) % 7;
       }
   else
       for (i = 1900; i > year; i--)
       {   weekday = (weekday - 365);
           if (IsLeapYear(i)) weekday = (weekday + 1) % 7;
       }
       for (i = 1; i < month; i++)
           weekday = (weekday + MonthDays(i, year)) % 7;
   return (weekday);
}
```

另外要修改的函数是 GetYearFromUser( )，此函数中不再进行年份判断。可修改如下：

```
int GetYearFromUser(void)
{   int year;
    while （1）
    {   printf("请输入要显示的年份：");
        scanf("%d", &year);
        if (year>= 0)
            return (year);
        printf("输入年份应该为正数!\n");
     }
}
```

7.6　利用库函数 rand( )与 n 进行模运算即 rand( )% n，可以得到 0 到 n–1 的随机数。但是得到的结果随机性还不够强。请编写程序，设计一种能利用 rand( )得到指定范围内某一随机数的方法。

解： 随机函数 int rand(void)的功能是生成 0 到 RAND_MAX 之间的伪随机数，RAND_MAX 的值依赖于不同的计算机系统。定义 rand( )的头文件是 stdlib.h。

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void Randmize(void);
int RandomInteger(int low, int high);
```

```
void main()
{   int low, high;
    printf("请输入要产生的随机数范围：");
    scanf("%d,%d", &low, &high);
    Randmize();
    printf("在%d 和%d 之间的随机数是：%d\n",low, high, RandomInteger(low,
          high));
}
int RandomInteger(int low, int high)
{   int k;
    double d;
    d =(double)rand()/((double)RAND_MAX + 1);
    k =(int)(d*(high-low+1));
    return(low + k);
}
void Randmize(void)
{   srand((int) time(NULL)); /* srand()为 rand()生成的随机数序列设置起点 */
}
```

程序运行情况如下：

```
请输入要产生的随机数范围：10,50↙
在 10 和 50 之间的随机数是：34
```

7.7 设计头文件将习题 7.6 中找到的随机数生成程序加入到自定义库中。

解：

```
/* 文件名: random.h */
#ifndef _random_h
#define _random_h
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void Randmozie(void);       /* Randmozie()设置随机种子的值为不可预测 */
int RandomInteger(int low, int high); /* RandomInteger()生成指定范围的随机数 */
#endif
```

7.8 编写一个程序显示从 52 张扑克牌中随机抽取一张牌的名字（包括它的花色）。

解：每张牌可能的等级为 A，2，3，4，5，6，7，8，9，10，J，Q，K。可能的花色为梅花（clubs），方片（diamonds），红桃（hearts），黑桃（spades）。调用习题 7.5 和习题 7.6 中设计的随机库函数即可。

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void Randmize(void);
int RandomInteger(int low, int high);
void main()
{   int level, color;
    Randmize();
    level = RandomInteger(1, 13);
    color = RandomInteger(1, 4);
    printf("随机抽取的扑克牌是：");
    switch (color)
    {   case 1: printf("黑桃"); break;
        case 2: printf("红桃"); break;
```

```
            case 3: printf("梅花"); break;
            case 4: printf("方片"); break;
        }
        switch (level)
        {   case 1: printf("A\n"); break;
            case 2:
            case 3:
            case 4:
            case 5:
            case 6:
            case 7:
            case 8:
            case 9:
            case 10: printf("%d\n", level); break;
            case 11: printf("J\n");break;
            case 12: printf("Q\n");break;
            case 13: printf("K\n");break;
        }
    }
    int RandomInteger(int low, int high)
    {   int k;
        double d;
        d = (double) rand() / ((double) RAND_MAX + 1);
        k = (int) (d * (high - low + 1));
        return (low + k);
    }
    void Randmize(void)
    {   srand((int) time(NULL));
    }
```

程序运行情况如下：

```
随机抽取的扑克牌是：黑桃 8
```

7.9　设计一个自定义函数库，以解决每月分期付款问题。公式如下：

$$m = \frac{i \times p}{1-(i+1)^{-12y}}$$

其中，m 是每月还款额，i 是每月利率（小数表示），p 是本金，y 是贷款年限。自定义库中应包含以下函数：(1) 在给定 p、i 和 y 时，计算每月还款额函数（要求 m 是整数，计算还款额时向下取整）；(2) 在给定固定利率、每月付款额和贷款期限时，计算可以借出的最大本金函数；(3) 输出贷款分期偿还表函数，将偿还次数、偿还金额、偿还利息、偿还本金，以及剩余金额分列显示。

解：这里仅给出前面两项的解答。分期付款常见的还款方式是等额本息还款和等额本金还款，题目中的公式是等额本息还款。等额本息还款的特点是每个月还款额度不变。

```
#include <math.h>
int CalcM(int principal, double interest_rate, int year)
{   int m;
    m=(int)interest_rate*principal/(1-pow((interest_rate+1),
      (double)-12*year) );
    return (m);
}
```

```
int CalcP(double interest_rate, int m, int year)
{   int principal;
    principal = (int) m * (1 - pow((interest_rate + 1),
                (double)-12 * year)) / interest_rate ;
    return (principal);
}
```

7.10 请以菜单组织模块法设计程序，解决如下问题：某班期末考试科目为数学（MT）、英语（EN）和物理（PH），有最多不超过 30 人参加考试，要求：

(1) 计算每个学生的总分和平均分；

(2) 按总分成绩由高到低排出成绩的名次；

(3) 打印出名次表，表格内包括学生编号、各科分数、总分和平均分；

(4) 任意输入一个学号，能够查找出该学生在班级中的排名及其考试分数。

解：

```
#include <stdio.h>
#include <stdlib.h>
#define STU 30
#define COURSE 3
void Input(long num[],int score[][COURSE],int n);
void GetSumAver(int score[][COURSE],int n,int sum[],float aver[]);
void Sort(long num[],int score[][COURSE],int n,int sum[],float aver[]);
void Print(long num[],int score[][COURSE],int n,int sum[],float aver[]);
int Search(long num[], int n, long x);
void main()
{   int     n, score[STU][COURSE], sum[STU], pos;
    long    num[STU], x;
    float   aver[STU];
    int     choice;
    do
    {   /* 显示主控菜单 */
        printf("       ***** 学生成绩管理系统 *****\n");
        printf("========================================\n");
        printf("  1. 成绩输入              2. 打印成绩\n");
        printf("  3. 排序                  4. 成绩查询\n");
        printf("          0. 退出系统\n");
        printf("========================================\n");
        printf("          请选择（0--4）: ");
        scanf("%d", &choice);
        switch(choice)
        {   case 1:
                printf("请输入学生人数 n(n<=30):");
                scanf("%d", &n);            /* 输入参加考试的学生人数 */
                printf("按学号 NO 数学 MT 英语 EN 物理 PH 的顺序输入信息\n");
                Input(num, score, n);             /* 输入学生成绩 */
                GetSumAver(score, n, sum, aver); /* 计算总分和平均分 */
                break;
            case 2:
                printf("成绩表:\n");
                Print(num, score, n, sum, aver);
                break;
            case 3:
                Sort(num, score, n, sum, aver); /* 排名次 */
                printf("按总分成绩，降序排列:\n");
```

```
                Print(num, score, n, sum, aver);
                break;
            case 4:
                printf("请输入要查找的学生的学号:");
                scanf("%ld", &x);          /* 以长整型格式输入待查找学生的学号 */
                pos = Search(num, n, x); /* 名次查询 */
                if (pos != -1)
                {   printf("position:\tNO\tMT\tEN\tPH\t SUM\t AVER\n");
                    printf("%8ld\t%4ld\t%4d\t%4d\t%4d\t%5d\t%5.0f\n",
                    pos+1,num[pos],score[pos][0],score[pos][1],
                    score[pos][2],sum[pos],aver[pos]);
                }
                else
                    printf("Not found!\n");
                break;
            case 0:
                exit(0);                /* 退出， 返回操作系统 */
            default:
                break;
        }
    } while (choice);
}
/* 函数功能： 输入某班学生期末考试三门课程成绩
   函数参数： 长整型数组 num，存放学生学号
              整型数组 score，存放学生成绩
              整型变量 n，存放学生人数
*/
void Input(long num[], int score[][COURSE], int n)
{   int  i, j;
    for (i=0; i<n; i++)
    {   scanf("%ld", &num[i]);
        for (j=0; j<COURSE; j++)
            scanf("%d", &score[i][j]);
    }
}
/* 函数功能： 计算每个学生的总分和平均分
   函数参数： 整型数组 score，存放学生成绩
              整型变量 n，存放学生人数
              整型数组 sum，计算得到的每个学生的总分
              实型数组 aver，计算得到的每个学生的平均分
*/
void GetSumAver(int score[][COURSE], int n, int sum[], float aver[])
{   int i, j;
    for (i=0; i<n; i++)
    {   sum[i] = 0;
        for (j=0; j<COURSE; j++)
            sum[i] = sum[i] + score[i][j];
        aver[i] = (float)sum[i] / COURSE;
    }
}
/* 函数功能： 按总分成绩由高到低排出成绩的名次
   函数参数： 长整型数组 num，存放学生学号
              整型数组 score，存放学生成绩
```

```
                整型变量 n，存放学生人数
                整型数组 sum，存放每个学生的总分
                实型数组 aver，存放每个学生的平均分
*/
void Sort(long num[],int score[][COURSE], int n, int sum[], float aver[])
{   int     i, j, k, m;
    int     temp1;
    long    temp2;
    float   temp3;
    for (i=0; i<n-1; i++)
    {   k = i;
        for (j=i+1; j<n; j++)
            if (sum[j] > sum[k])  k = j;
        if (k != i)
        {   temp1 = sum[k];  sum[k] = sum[i];  sum[i] = temp1;
            temp2 = num[k];  num[k] = num[i];  num[i] = temp2;
            temp3 = aver[k];  aver[k] = aver[i];  aver[i] = temp3;
            for (m=0; m<COURSE; m++)
            {   temp1 = score[k][m];
                score[k][m] = score[i][m];
                score[i][m] = temp1;
            }
        }
    }
}
/* 函数功能：  打印出名次表，表格内包括学生编号、各科分数、总分和平均分
   函数参数：  长整型数组 num，存放学生学号
               整型数组 score，存放学生成绩
               整型变量 n，存放学生人数
               整型数组 sum，存放每个学生的总分
               实型数组 aver，存放每个学生的平均分
*/
void Print(long num[], int score[][COURSE], int n, int sum[], float aver[])
{   int  i, j;
    printf("  NO \t|   MT \t  EN \t PH \t SUM \t AVER\n");
    printf("-------------------------------------------------------\n");
    for (i=0; i<n; i++)
    {   printf("%4ld\t| ", num[i]);
        for (j=0; j<COURSE; j++)
            printf("%4d\t", score[i][j]);
        printf("%5d\t%5.0f\n", sum[i], aver[i]);
    }
}
/* 函数功能：  在学号数组中顺序查找学生的学号
   函数参数：  长整型数组 num，存放学生学号
               整型变量 n，存放学生人数
               长整型变量 x，存放待查找学生的学号
               函数返回值：找到时，返回学生学号在学号数组中的下标位置
               否则，返回值-1
*/
int Search(long num[], int n, long x)
{   int  i;
    for (i=0; i<n; i++)
```

```
    if (num[i] == x)  return(i);
    return (-1);
}
```

程序运行情况如下：

```
       ***** 学生成绩管理系统 *****
==========================================
  1. 成绩输入                2. 打印成绩
  3. 排序                    4. 成绩查询
          0. 退出系统
==========================================
          请选择（0--4）：1↙
请输入学生人数 n(n<=30):4↙
按学号 NO 数学 MT 英语 EN 物理 PH 的顺序输入信息
08001  75 85 63↙
08002  72 84 76↙
08003  61 95 86↙
08004  88 69 77↙
       ***** 学生成绩管理系统 *****
==========================================
  1. 成绩输入                2. 打印成绩
  3. 排序                    4. 成绩查询
          0. 退出系统
==========================================
          请选择（0--4）：2↙
成绩表:
  NO    |   MT    EN      PH      SUM      AVER
-----------------------------------------------------
8001    |   75    85      63      223      74
8002    |   72    84      76      232      77
8003    |   61    95      86      242      81
8004    |   88    69      77      234      78
       ***** 学生成绩管理系统 *****
==========================================
  1. 成绩输入                2. 打印成绩
  3. 排序                    4. 成绩查询
          0. 退出系统
==========================================
          请选择（0--4）：3↙
按总分成绩，降序排列:
  NO    |   MT    EN      PH      SUM      AVER
-----------------------------------------------------
8003    |   61    95      86      242      81
8004    |   88    69      77      234      78
8002    |   72    84      76      232      77
8001    |   75    85      63      223      74
       ***** 学生成绩管理系统 *****
==========================================
  1. 成绩输入                2. 打印成绩
  3. 排序                    4. 成绩查询
          0. 退出系统
==========================================
          请选择（0--4）：0↙
```

第 8 章　构造数据类型

8.2　有 10 个学生，每个学生的数据包括学号、姓名和三门课的成绩。用键盘输入 10 个学生的数据，要求打印出每个学生三门课的平均成绩，以及最高分的学生的数据（学号、姓名、三门课的成绩和平均分数）。

解：

```
#include <stdio.h>
#include <string.h>
typedef struct                  /*定义学生结构体类型，并把该类型名自定义为 STUDENT*/
{   char name[20];
    char sno[20];
    int score_1;
    int score_2;
    int score_3;
    double score_all;
} STUDENT;
#define N 10
STUDENT students[N]={0};    /* 定义结构体数组并把所有元素初始化为零 */
void main()
{   int i,max;
    for (i=0;i<N;i++)       /* 输入 10 个学生的相关信息 */
    {   printf("Input #%d Student's Name:",i+1);
        scanf("%s",students[i].name);
        printf("Input #%d Student's SNO:",i+1);
        scanf("%s",students[i].sno);
        printf("Input #%d Student's Score_1:" ,i+1);
        scanf("%d",&students[i].score_1);
        printf("Input #%d Student's Score_2:" ,i+1);
        scanf("%d",&students[i].score_2);
        printf("Input #%d Student's Score_3:" ,i+1);
        scanf("%d",&students[i].score_3);
        students[i].score_all=students[i].score_1+
          students[i].score_2+students[i].score_3;  /* 计算各个学生的总分 */
    }
    max=0;
    for(i=0;i<N;i++)                    /* 输出所有学生的信息包括平均分 */
{   printf("%s %s %d %d %d %lf\n",students[i].name, students[i].sno,
    students[i].score_1, students[i].score_2, students[i].score_3,
    students[i].score_all/3);
    if (students[max].score_all<students[i].score_all)    /*求最高分 */
        max=i;
    }
    /* 打印出最高分的学生的数据 */
    printf("top=%s %s %d %d %d %lf \n",students[max].name,
        students[max].sno,students[max].score_1,students[max].score_2,
        students[max].score_3,students[max].score_all/3);
}
```

8.3　定义一个结构体变量（包括年、月、日）。写一个函数 days，实现计算该日期在本年中是第几天，由主函数将年、月、日传给 days 函数，计算后将日子数传回主函数输出。注意闰年问题。

解：解题思路如下：正常年份每个月中的天数是已知的，只要给出日期，算出该日在本年中是第几天是不困难的。如果是闰年且月份在 3 月或 3 月以后时，应再增加 1 天。闰年的规则是：年份能被 4 和 400 整除但不能被 100 整除，如 2000 年、2004 年、2008 年是闰年，2100 年、2005 年不是闰年。

方法一：

```
#include <stdio.h>
struct date                          /* 定义日期结构体类型，并定义结构体变量 d */
    {   int year;
        int month;
        int day;
    }d;
int  days(struct date date1);   /* days 函数的声明 */
void main()
{   printf("input year,month,day:");
    scanf("%d,%d,%d",&d.year,&d.month,&d.day);
    printf("%d/%d is the %dth day in %d.\n",d.month,d.day,days(d),d.year);
}
int  days(struct date date1)
{   int sum;
    switch(date1.month) /* 事先设置好 1 到 month-1 月的天数，加上当月的天数即可求解 */
    {   case 1: sum=date1.day;    break;
        case 2: sum=date1.day+31;  break;
        case 3: sum=date1.day+59;  break;
        case 4: sum=date1.day+90;  break;
        case 5: sum=date1.day+120; break;
        case 6: sum=date1.day+151; break;
        case 7: sum=date1.day+181; break;
        case 8: sum=date1.day+212; break;
        case 9: sum=date1.day+243; break;
        case 10: sum=date1.day+273; break;
        case 11: sum=date1.day+304; break;
        case 12: sum=date1.day+334; break;
    }
/* 判断若为闰年且月份在 3 月或 3 月以后时，应再增加 1 天 */
if((date1.year%4==0 && date1.year%100!=0
    ||date1.year%400==0) && date1.month>=3)
  sum+=1;
 return(sum);
}
```

程序运行情况如下：

```
input year,month,day:2008,8,8
8/8 is the 221th day in 2008.
```

在本程序中，days 函数参数为结构体 date 类型，在主函数的第二个 printf 语句的输出项中有一项为 days(d)，调用 days 函数，实参为结构体变量 d。通过虚实结合，将实参 d 中各成

员的值传递给 date1 中各相应成员。在 days 函数中检验其中 month 的值，根据它的值计算出天数 sum，将 sum 的值返回主函数输出。

方法二：

```
#include <stdio.h>
struct y_m_d
{   int year;
    int month;
    int day;
}date;
void main()
{   int days(int year,int month,int day);
    int days(int,int,int);
    int day_sum;
    printf("input year,month,day:");
    scanf("%d,%d,%d",&date. year,&date.month,&date.day);
    day_sum=days(date.year,date.month,date.day);
    printf("%d / %d is the %dth day in %d.\n",
            date.month,date.day,day_sum,date.year);
}
int days(int year,int month,int day)
{   int day_sum,i;
    int day_tab[13]={0,31,28,31,30,31,30,31,31,30,31,30,31};
                                        /* 1~12 各个月的天数 */
    day_sum=0;
    for (i=1;i<month;i++)               /* 计算 1 到 month-1 月的天数 */
        day_sum+=day_tab[i];
    day_sum+=day;
    if ((year%4==0 && year%100!=0 || year%4==0) && month>=3)
        day_sum+=1;
    return(day_sum);
}
```

程序运行情况如下：

```
input year,month,day:2008,8,8
8/8 is the 221th day in 2008.
```

在本程序中，days 函数的参数为结构体变量的成员 year、month、day，而不是整个结构体变量。

可以看到，在定义了结构体变量后，使用时有不同的方法。

8.4 利用结构体类型分别写出复数的加、减运算函数，并在主函数中调用这些函数。

解：

```
#include <stdio.h>
struct data             /* 定义复数结构体 */
{   float real;         /* 复数的实部 */
    float imag;         /* 复数的虚部 */
};
void subtract (struct data x, struct data y);
void add(struct data x, struct data y);
void main()
{   struct data z1,z2;
```

```
    printf("请输入第一个复数的实部和虚部：");
    scanf("%f%f",&z1.real,&z1.imag);
    printf("请输入第二个复数的实部和虚部：");
    scanf("%f%f",&z2.real,&z2.imag);
    add(z1,z2);
    subtract(z1,z2);
}
void add(struct data x, struct data y)          /* 定义求复数加的函数 */
{   struct data z;
    z.real=x.real+y.real;
    z.imag=x.imag+y.imag;
    if(z.real!=0)
        if (z.imag >0)
            printf("复数的和是：%f+%fi\n",z.real,z.imag);
        else if(z.imag <0)
                printf("复数的和是：%f%fi\n",z.real,z.imag);
        else
                printf("复数的和是：%f\n",z.real);
    else
        printf("复数的和是：%fi\n",z.imag );
}
void subtract (struct data x, struct data y)   /* 定义求复数减的函数 */
{  struct data z;
   z.real=x.real-y.real;
   z.imag=x.imag-y.imag;
   if(z.real !=0)
      if (z.imag >0)
           printf("复数的差是：%f+%fi\n",z.real,z.imag);
      else if(z.imag <0)
           printf("复数的差是：%f%fi\n",z.real,z.imag);
      else
           printf("复数的差是：%f\n",z.real);
    else
        printf("复数的差是：%fi\n",z.imag );
}
```

程序运行情况如下：

```
请输入第一个复数的实部和虚部：12  36
请输入第二个复数的实部和虚部：23 36
复数的和是：35.000000+72.000000i
复数的差是：-11.000000
```

8.5　定义结构体，要求利用结构体变量求解两个复数之积。

解：

```
#include <stdio.h>
struct data                                     /* 定义复数结构体 */
{   float real;
    float imag;
};
void mul(struct data x, struct data y);     /* mul 函数的声明 */
void main()
{   struct data z1,z2;
    printf("请输入第一个复数的实部和虚部：");
```

```
    scanf("%f%f",&z1.real,&z1.imag);
    printf("请输入第二个复数的实部和虚部：");
    scanf("%f%f",&z2.real,&z2.imag);
    mul(z1,z2);
}
void mul(struct data x, struct data y)      /* 定义求复数乘积的 mul 函数 */
{   struct data z;
    z.real=x.real*y.real-x.imag*y.imag;
    z.imag=x.real*y.imag+y.real*x.imag;
    if(z.real !=0)
        if (z.imag >0)
            printf("复数的积是：%f+%fi\n",z.real,z.imag);
        else if(z.imag <0)
            printf("复数的积是：%f%fi\n",z.real,z.imag);
        else
            printf("复数的积是：%f\n",z.real);
    else
        printf("复数的积是：%fi\n",z.imag );
}
```

程序运行情况如下：

```
请输入第一个复数的实部和虚部：2 3
请输入第二个复数的实部和虚部：4 6
复数的积是：-10.000000+24.000000i
```

8.6 已知某班有 5 名同学，成绩如下：

学号	姓名	性别	C 语言	数据结构	汇编语言	计算机原理
1001	刘民	男	67	58	78	98
1002	梁天	女	86	98	88	89
1003	郑丽	女	89	82	69	95
1004	张三	男	78	64	56	78
1005	赵吴	男	65	63	25	77

要求：(1) 用结构体数组处理；

(2) 数据的初始化和输出应在主函数中进行；

(3) 编写函数 sort，按学生的总成绩大小排序。

解：

```
#include <stdio.h>
#include <string.h>
typedef struct          /* 定义学生结构体类型，并把该类型名自定义为 STUDENT */
{   char sno[20];
    char name[20];
    char sex[2];
    int score[4];
    int score_all;
} STUDENT;
#define N 5
STUDENT students[N]={0};
void sort (STUDENT stu[],int n);
void main()
```

```
{   int i,j;
    for (i=0;i<N;i++)
    {   printf("请输入第%d 个学生的学号:",i+1);
        scanf("%s",students[i].sno);
        printf("请输入第%d 个学生的名字:",i+1);
        scanf("%s",students[i].name);
        printf("请输入第%d 个学生的性别:",i+1);
        scanf("%s",students[i].sex);
        printf("请输入第%d 个学生的 4 门课程的分数:",i+1);
        for(j=0;j<4;j++)                    /* 输入 4 门课的分数*/
        {   scanf("%d",&students[i].score[j]);
            students[i].score_all+=students[i].score[j];/* 求4门课总分 */
        }
    }
    sort(students,N);                       /* 调用 sort 函数按总分排序 */
    for(i=0;i<N;i++)
    printf(" %s %s %s %d %d %d %d %d\n", students[i].sno,students[i].name,
        students[i].sex,students[i].score[0],students[i].score[1],
        students[i].score[2],students[i].score[3],students[i].score_all);
}
void sort (STUDENT stu[],int n)             /* 定义按总分排序的 sort 函数 */
{   int i,j;
    STUDENT temp;
    for(i=1;i<=n-1;i++)                     /* 冒泡排序法 */
        for(j=0;j<=n-i-1;j++)
            if (stu[j].score_all<stu[j+1].score_all)
            {   temp=stu[j];stu[j]=stu[j+1];stu[j+1]=temp; }
}
```

程序运行情况如下：

```
请输入第 1 个学生的学号:1001↙
请输入第 1 个学生的名字:刘民↙
请输入第 1 个学生的性别:男↙
请输入第 1 个学生的 4 门课程的分数:67 58 78 98↙
请输入第 2 个学生的学号:1002↙
请输入第 2 个学生的名字:梁天↙
请输入第 2 个学生的性别:女↙
请输入第 2 个学生的 4 门课程的分数:86 98 88 89↙
请输入第 3 个学生的学号:1003↙
请输入第 3 个学生的名字:郑丽↙
请输入第 3 个学生的性别:女↙
请输入第 3 个学生的 4 门课程的分数:89 82 69 95↙
请输入第 4 个学生的学号:1004↙
请输入第 4 个学生的名字:张三↙
请输入第 4 个学生的性别:男↙
请输入第 4 个学生的 4 门课程的分数: 78 64 56 78↙
请输入第 5 个学生的学号:1005↙
请输入第 5 个学生的名字:赵吴↙
请输入第 5 个学生的性别:男↙
```

```
请输入第 5 个学生的 4 门课程的分数: 65 63 25 77↙
1002 梁天 男 86 98 88 89  361
1003 郑丽 女 89 82 69 95  335
1001 刘民 女 67 58 78 98  301
1004 张三 男 78 64 56 78  276
1005 赵吴 男 65 63 25 77  230
```

8.7 定义公司员工工资结构体类型，描述工资情况的数据项有发放年月、工资号、姓名、基本工资、活动工资、津贴、税费和实发工资等。用工资结构体类型定义工资结构体数组，用于存储员工工资信息。从键盘输入 N 名员工的发放年月、工资号、姓名、基本工资、活动工资、津贴和税费等数据项，计算每名员工的实发工资，然后将所用员工工资的数据输出。

解：实发工资的计算公式为实发工资=基本工资+活动工资+津贴–税费。

```
#include <stdio.h>
#define N 10
struct Employee                 /* 定义工资结构类型 */
{   long yearmonth ,id;
    char name[16];
    float base_sa,act_sa,subsidy,tax,fact_sa;
};
void main()
{   struct Employee e[N];       /* 定义结构体数组 */
    int i;
    printf("请按如下格式输入%d 个员工的信息：\n",N);
    printf("yearmonth id name base_sa act_sa subsidy tax\n");
    for(i=0;i<N;i++)            /* 输入每个员工的信息 */
    {   printf("请输入%dth 员工的信息：\n",i+1);
        scanf("%ld%ld%s%f%f%f%f",&e[i].yearmonth,&e[i].id,e[i].name,
            &e[i].base_sa,&e[i].act_sa,&e[i].subsidy,&e[i].tax);
    }
    for(i=0;i<N;i++)            /* 计算每名员工的应发工资 */
        e[i].fact_sa= e[i].base_sa+ e[i].act_sa+ e[i].subsidy- e[i].tax;
            /* 输出表头 */
    printf("%-10s%-10s%-16s%-8s%-8s%-8s%-8s%-8s\n","yearmonth","Id",
            "Name","Base_sa","Act_sa",,"subsidy","Tax","Fact_sa");
    for(i=0;i<N;i++)            /* 输出员工工资情况 */
         printf("%-10ld%-10ld%-16s%-8.1f%-8.1f%-8.1f%-8.1f%-8.1f\n",
            e[i].yearmonth ,e[i].id, e[i].Name, e[i].Base_sa,
            e[i].Act_sa, e[i].subsidy, e[i].Tax, e[i].Fact_sa);
}
```

第 9 章 位 运 算

9.1 请叙述位段的概念。

解：位段又称为位域或位组，是 C 语言提供的一种数据结构，就是将字节以位为单位划分为不同的区域，每个区域长度小于 8 位，定义时说明位数和域名，如此，就可以将存储空间小于一个字节的多个信息存储到一个字节中，使存储空间得到充分利用；在程序中能按域名对位段进行操作。

在 C 语言中，位段是以位为单位来定义成员长度的特殊结构体，每个成员所占位数小于 8（bit）。

9.2　设计一个程序，用位运算实现给出一个数的原码，求该数的补码。

解：提示：正数的补码与其原码形式相同，负数的补码等于该数的反码加 1。

```
#include <stdio.h>
unsigned int tranbit(unsigned int);
void main()
{   unsigned int a;
    printf("Please input a hex number:");
    scanf("%x",&a);
    printf("result:%x\n",tranbit(a));
}
unsigned int tranbit(unsigned int n)
{   if(n&1000000000000000)
        return((~n+1)|1000000000000000);
    else
        return(n);
}
```

程序运行情况如下：

```
Please input a hex number:8003↙  （-3 原码：10000 0000 0000 0011）
result:fffd    (-3 补码：1111 1111 1111 1101)
```

再运行一次如下：

```
Please input a hex number:8007↙  （-7 原码：10000 0000 0000 0111）
result:fff9    (-7 补码：1111 1111 1111 1001)
```

9.3　设计一个成员为位段的结构体类型，内含位段 a（3 位），b（6 位），c（3 位），d（3 位），并测试此类型数据的长度。

解：
```
#include <stdio.h>
void main()
{   struct pack_data
    {   unsigned a :3;
        unsigned  :0;
        unsigned b :6;
        unsigned c :3;
        unsigned d :3;
    }bs;
    printf("size of bit:%d bytes\n",sizeof(bs)); /* sizeof()求变量字节函数 */
}
```

程序运行情况如下：

```
size of bit:3 bytes
```

9.4　编写将一个字节的高 4 位改为 1010 的函数。

解：提示：或运算：与 1 相或对应位置 1，与 0 相或对应位不变；

　　与运算：与 1 相与对应位不变，与 0 相与对应位置 0。

```
#include <stdio.h>
void main()
```

```
{   char a;
    scanf("Please input a byte hex number:%x",&a);
    printf("%x,%x",a,(a|0xa0)&0xaf);
}
```

9.5 编写一个函数，把 16 位的整型变量 a 中的低字节与变量 b 中的高字节合并放入变量 c 中，并输出 c。

解：

```
#include <stdio.h>
void main()
{   int a,b,c;
    printf("Please input a and b:\n");
    scanf("%d%d",&a,&b);
    printf("a=%d(%xH),b=%d(%xH)",a,a,b,b);
    c=a&0xff00;     /* 用与运算提取 a 的高字节 */
    c=c|(b&0x00ff);  /* 用b和0x00ff的与运算提取b的低字节,再与c作或运算合并高低字节 */
    printf("\nc=%d(%xH)",c,c);
}
```

程序运行情况如下：

```
Please input a and b:
900↙
500↙
a=900(384H),b=500(1f4H)          /* a 高字节为 03H，b 低字节为 f4H */
c=1012(3f4H)
```

9.6 编写一个函数，用它测定一个整数的最高位是 0 还是 1，如果是 0，则输出“正数”，如果是 1，则输出“负数”。

解：

```
#include <stdio.h>
int test(char);
void main()
{   char a;
    printf("Please input a number:");
    scanf("%d",&a);
    if (test(a))
        printf("\npositive number");
    else
        printf("\nnegative");
}
int test(char n)
{   if(n&10000000)          /* 和 10000000 做与运算 */
        return (0);         /* 最高位为 1,结果>0，返值 0 */
    else
        return(1);          /* 否则结果<0，返值 1 */
}
```

9.7 编写一个函数，用移位的方法使一个整数 a 乘 2，并输出结果。

解：

```
#include <stdio.h>
int shift(int);
void main()
{   int a;
```

```
    printf("Please input a number:");
    scanf("%d",&a);
    printf("\n%d",shift(a));
}
int shift(int n)
{   return(n<<1);
}
```

程序运行情况如下：

```
Please input a number:553↙
1106
```

9.8　一个数与 0 进行按位异或，结果是什么？编写一个程序实现之，并分析输出结果。

解：

```
#include <stdio.h>
int shift(int);
void main()
{   int a;
    printf("Please input a number:");
    scanf("%d",&a);
    printf("\n%d",a^0);
}
```

程序运行情况如下：

```
Please input a number:5236↙
5236
```

再次运行：

```
Please input a number:12↙
12
```

分析：可见，任何数与 0 异或其值不变。

9.9　编写一个函数，使一整数 a 的二进制表示形式中的高 3 位（第 5~7 位）翻转。

解：

```
#include <stdio.h>
unsigned char turn(unsigned char);
void main()
{   unsigned char a;
    printf("Please input a number(<256 ):");
    scanf("%d",&a);
    printf("%d=%xH",a,a);          /* 显示输入数的十进制和十六进制形式 */
    printf("\nturn: %d=%xH ",turn(a),turn(a));
                                   /* 翻转后十进制和十六进制形式 */
}
unsigned char turn(unsigned char n)
{   /* 编程依据：与 1 异或其值翻转，与 0 异或其值不变。*/
    return (n^0xe0);              /* 0xe0 的二进制表示为 1110 0000。*/
}
```

程序运行情况如下：

```
Please input a number(<256):197↙
197=c5H
Turn:37=25H
```

再运行一次如下：

```
Please input a number(<256):3↙
3=3H
Turn:277=e3H
```

9.10 一种文本字符的加密方式是,对每一个字节字符，将其高 4 位和低 4 位互换。根据此加密方式，编写两个函数分别实现对字符串的加密和解密。

解：分析：加密是将高 4 位和低 4 位互换后合并，解密是逆过程，可见仍是将高 4 位和低 4 位互换后合并，因此，加密和解密可共用一个子函数，编程如下：

```
#include <stdio.h>
void jmjm(char[]);
void main()
{   char a[60];
    printf("Please input a character string: ");
    gets(a);
    jmjm(a);                /* 加密 */
    printf("encrypt:%s",a);
    jmjm(a);                /* 解密 */
    printf("\ndecrypt:%s",a);
}
void jmjm(char n[])
{   int i;
    char h,l;
    for(i=0;n[i]!='\0';i++)
    {   h=(n[i]&0xf0)>>4;   /* 取高 4 位且右移 4 位,高 4 位变低 4 位 */
        l=(n[i]&0x0f)<<4;   /* 取低 4 位左移 4 位, 低 4 位变高 4 位 */
        n[i]=l|h;           /* 互换后组合 */
    }
}
```

程序运行情况如下：

```
Please input a character string:dyf6yrccc↙
encrypt:Fùfcù'666
decrypt:dyf6yrccc
```

再运行一次如下：

```
Please input a character string:fg7865t↙
encrypt:fvsâcSG
decrypt:fg7865t
```

第 10 章 文 件

10.4 编写程序，实现从键盘输入一些字符（以“#”结束），并存入一个磁盘文件。

解：

```
#include <stdio.h>
void main()
{  FILE  *fp ;
   char ch ;
   if ((fp = fopen("d:\\aa.txt","w")) == NULL) /* 打开（建立）磁盘文件 */
```

```
    {   printf (" Cannot open file\n ");
        exit (0) ;                          /* 若打开出错，则返回操作系统 */
    }
    while ((ch = getchar())!= '#')          /* 从键盘输入一串字符,#号结束 */
        fputc(ch, fp) ;                     /* 写入磁盘文件 */
    printf(" \nO.K!\n ") ;                  /* 屏幕提示操作完成 */
    fclose(fp);
}
```

程序运行情况如下：

```
How are you?# ↙
O.K!
```

可以在 D 盘下找到建立的磁盘文件 aa.txt，内容为输入内容：How are you?

10.5　将主教材中例 10.3 所建立的磁盘文件内容显示到屏幕上。

解：

```
#include <stdio.h>
void main()
{   FILE *fpb;                              /* 定义文件指针：fpb */
    char ch;
    fpb=fopen("d:\\cc\\scor.txt","r");      /* 为“读”打开指定文件 */
    if(fpb == NULL)                         /* 若打开文件时发生错误 */
    {   printf("can\'t open this file");    /* 在显示器上显示出错信息 */
        exit(0);                            /* 退出程序 */
    }
    while((ch = fgetc(fpb))!= EOF)          /* 逐个字符输出，直到文本结束 */
        putchar(ch) ;
    fclose(fpb);                            /* 关闭文件 */
}
```

说明：d:\cc\scor.txt 是主教材中例 10.3 建立的磁盘文件。做本题前，须运行例 10.3 建立起 d:\cc\scor.txt 文件，也可用写字板建立此文件，若文件不存在，会出错。

10.6　编写程序，实现从键盘输入一些字符（以 “#”结束），进行加密后（每个字符的 ASCII 码加 4）写入新建磁盘文件，然后从磁盘将信息读出并解密还原后显示在屏幕上。

解：

```
#include <stdio.h>
void main()
{   FILE  *fp,*fpd;
    char ch ;
    if ((fp = fopen("d:\\wjt96.txt","w"))==NULL) /* 打开（建立）磁盘文件 */
    {   printf (" Cannot open file\n ");
        exit (0) ;                          /* 若打开出错，则返回操作系统 */
    }
    printf("Please input a character string(# end):\n");
    while ((ch = getchar())!= '#')
    {   fputc(ch+4,fp);                     /* 加密后写入文件 */
        putchar(ch+4);                      /* 显示加密后的字串到屏幕 */
    }
    printf(" \nO.K!\n ") ;                  /* 屏幕提示写文件完成 */
    fclose(fp);
    fpd=fopen("d:\\wjt96.txt","r");         /* 为“读”打开指定文件 */
```

```
        if(fpd == NULL)                    /* 若打开文件时发生错误 */
        {   printf("can\'t open this file");      /* 在显示器上显示出错信息 */
            exit(0);                       /* 退出程序 */
        }
    while((ch = fgetc(fpd))!= EOF)         /* 从文件逐个获取字符直到文本结束 */
        putchar(ch-4) ;                    /* 解密后显示字符到屏幕 */
    fclose(fpd);                           /* 关闭文件 */
}
```

程序运行情况如下：

```
Please input a character string(# end):
Good jjjjj,,bb#↙
Kssh$nnnnn00ff
O.K!
Good jjjjj,,bb
```

10.7　参照主教材中例 10.8，请建立学生的成绩档案文件，要求每个学生保存如下信息：学号、姓名及 3 门功课成绩（score1、score2、score3）；从键盘输入 5 个学生的相关信息，并将其存入 xscj.dat 二进制数据文件。

解：

```
#include <stdio.h>
struct student                         /* 结构体类型声明 */
{   char  name[8];
    char  num[7];
    int score1;
    int score2;
    int score3;
};
void main()
{   struct student  stu[5];
    int i=0;
    FILE *fp=NULL;
    printf("Please input 5 records:\n");
    for(i=0;i<5;i++)                           /* 从键盘循环接收 5 名学生信息 */
        scanf("%s%s%d%d%d",&stu[i].name,&stu[i].num,&stu[i].score1,
                &stu[i].score2,&stu[i].score3);
    fp=fopen("d:\\xscj.dat","wb");             /* 用“写”方式打开二进制文件 */
    if(fp == NULL)                             /* 若打开文件时发生错误 */
    {   printf("can\'t open this file\n");     /* 在显示器上显示出错信息 */
        exit(0);                               /* 退出程序 */
    }
    for(i=0;i<5;i++)              /* 用循环方式调用 fwrite()将信息写入文件 */
        fwrite(stu+i,sizeof(struct student),1,fp);
    fclose(fp);
}
```

程序运行情况如下：

```
Please input 5 records:
Wang 2001 100 100 100↙
Liu 2002 80 80 80↙
```

```
Cheng 2003 85 85 85↙
Li 2008 95 95 95↙
Hao 2010 90 90 90↙
```

程序运行结束，可在 D 盘下找到 d:\xscj.dat 文件。

10.8　参照主教材中例 10.9 和例 10.11，将习题 10.7 所建文件“xscj.dat”中第 1,3,5 号学生的信息数据读出，并显示到屏幕。

解：

```
#include <stdio.h>
struct student                                  /* 结构体类型声明 */
{   char  name[8] ;
    char  num[7] ;
    int  score1;
    int  score2;
    int  score3;
};
void main()
{   struct student  stu={0};
    int i=0; FILE *fp=NULL;
    fp=fopen("d:\\xscj.dat","rb");              /* 用“读”方式打开二进制文件 */
    if(fp == NULL)                              /* 若打开文件时发生错误 */
    {   printf("can\'t open this file");       /* 在显示器上显示出错信息 */
        exit(0);                                /* 退出程序 */
    }
    printf("all records:\n");
    for(i=0;i<5;i++)
    {   fread(&stu,sizeof(struct student),1,fp); /* 一次读一个数据块 */
        printf("%s%s%d%d%d\n",stu.name,stu.num,stu.score1,stu.score1,
                stu.score1);
    }
    printf("\n\n1,3,5 records:\n");
    rewind(fp);                                 /* 文件读写位置指针回到文件头 */
    for(i=0;i<3;i++)
    {   fread(&stu,sizeof(struct student),1,fp); /*一次读一个数据块 */
        printf("%s%s%d%d%d\n",stu.name,stu.num,stu.score1,stu.score1,
                stu.score1);
        fseek(fp,sizeof(struct student),1);/* 移动读写位置指针跳过一条记录 */
    }
    fclose(fp);
}
```

程序运行情况如下：

```
all records:
wang 2001 100 100 100
liu 2002 80 80 80
cheng 2003 85 85 85
li 2008 95 95 95
hao 2010 90 90 90

1,3,5 records:
wang 2001 100 100 100
cheng 2003 85 85 85
hao 2010 90 90 90
```

10.9 利用习题 10.7 所建的“xscj.dat”文件，将学生的姓名、学号两项数据复制到一个新文件“student.txt”中。

解：

```
#include <stdio.h>
struct student                              /* 结构体类型声明 */
{  char  name[8] ;
   char  num[7] ;
};
void main()
{  struct student  stu={0};
   int i=0;
   FILE *fp1=NULL, *fp2=NULL;
   fp1=fopen("d:\\xscj.dat","rb");     /* 用“读”方式打开二进制文件 */
   if(fp1 == NULL)
   {   printf("can\'t open this file"); /* 在显示器上显示出错信息 */
       exit(0);
   }
   fp2=fopen("d:\\student.txt","w");   /* 用“写”方式打开文本文件 */
   if(fp2 == NULL)                     /* 若打开文件时发生错误 */
   {   printf("can\'t open this file"); /* 在显示器上显示出错信息 */
       exit(0);
   }
   for(i=0;i<5;i++)
   {   fread(&stu,sizeof(struct student),1,fp1);  /* 读姓名、学号信息 */
       printf("%s %s\n",stu.name,stu.num);        /* 显示姓名、学号 */
       fprintf(fp2,"%s %s\n",stu.name,stu.num);   /*将姓名、学号写入文件 */
       fseek(fp1,6,1);
   }
   fclose(fp1);
   fclose(fp2);
}
```

程序运行情况如下：

```
wang 2001
liu 2002
cheng 2003
li 2008
hao 2010
```

在 D 盘下，可找到“student.txt”文件。

10.10 有两个磁盘文件 A.dat 和 B.dat，各存放有一串字符，要求编一程序，将两个文件的内容读出并保存到 C.dat 文件中去。

提示：① 首先要用记事本建立两个数据文件：A.dat 和 B.dat 各存放一个任意字符串；

② 将 A.dat 和 B.dat 的内容写入文件 C.dat 时，一个用写方式，一个要用添加方式。

编程实现如下：

解：

```
#include <stdio.h>
void main()
{   char ch;
    FILE *fp1=NULL, *fp2=NULL, *fp3=NULL;
```

```
    fp1=fopen("d:\\A.dat","r");              /* 用“读”方式打开文件 A */
    if(fp1 == NULL)
    {   printf("can\'t open this file");    /* 在显示器上显示出错信息 */
        exit(0);
    }
    fp2=fopen("d:\\B.DAT","r");              /* 用“读”方式打开文件 B */
    if(fp2 == NULL)                          /* 若打开文件时发生错误 */
    {   printf("can\'t open this file");    /* 在显示器上显示出错信息 */
        exit(0);
    }
    fp3=fopen("d:\\C.dat","wa");             /* 用“写添加”方式打开文件 C */
    if(fp3 == NULL)
    {   printf("can\'t open this file");
        exit(0);
    }
    printf("read A.dat,write C.dat\n");
    while ((ch=fgetc(fp1))!=EOF)             /* 逐个字符读 A 文件 */
    {   putchar(ch);                         /* 逐个显示字符 */
        fputc(ch,fp3);                       /* 逐个字符写入 C 文件 */
    }
    fclose(fp1);
    printf("\n\nread B.dat,write C.dat\n");
    while ((ch=fgetc(fp2))!=EOF)
    {   putchar(ch);
        fputc(ch,fp3);
    }
    fclose(fp2);
    fclose(fp3);
    fp3=fopen("d:\\C.dat","r");
    if(fp3 == NULL)
    {   printf("can\'t open this file"); exit(0);  }
    printf("\n\nread C.dat \n");
    while ((ch=fgetc(fp3))!=EOF)    putchar(ch);
    fclose(fp3);
}
```

A.dat 文件内容如下：

```
abcdefg
rrrrrrr
```

B.dat 文件内容如下：

```
rrrrr ttttttt yyyyyyyy,,,yyyy
```

程序运行情况如下：

```
read A.dat,write C.dat
abcdefg
rrrrrrr

read B.dat,write C.dat
rrrrr ttttttt yyyyyyyy,,,yyyy
```

```
read C.dat
abcdefg
rrrrrrr
rrrrr ttttttt yyyyyyyy,,,yyyy
```

程序运行结束后，可在 D 盘下找到“C.dat”文件。

10.11 C 语言对文件的操作总是通过调用函数进行的，请总结本章所讲述的文件操作函数，将其分类填入如下表格，以便学习使用。

解：

分 类	函 数 名	功 能	使 用 方 式
打开文件	fopen( )	打开文件	文件指针 = fopen（文件名，操作方式码）;
关闭文件	fclose( )	关闭文件	fclose（文件指针）;
文件读写	fgetc( ),getc( )	从文件取一个字符	字符接收变量 = fgetc（文件指针）;
	fputc( ),putc( )	存一个字符到文件	fputc（输出字符或字符变量, 文件指针）;
	gets( )	从文件取一字符串	fgets（字串地址, 字符个数, 文件指针）;
	fputs( )	存一字符串到文件	fputs（字串地址, 文件指针）;
	fread( )	从文件取数据块	fread（输入块地址, 块字节数, 块数, 文件指针）;
	fwrite( )	存数据块到文件	fwrite（输出块地址, 块字节数, 块数, 文件指针）;
	fscanf( )	从文件按格式取数据	fscanf（文件指针，格式字符串，输入项列表）;
	fprintf( )	按格式存数据到文件	fprintf（文件指针，格式字符串，输出项列表）;
文件定位	fseek( )	移动文件读写位置指针	fseek（文件指针，位移量，起始点）;
	Rewind( )	移读写指针到文件头	rewind（文件指针）;
	ftell( )	获取当前读写指针位置	文件读写指针位置变量 = ftell（文件指针）;
文件状态测试	feof( )	文件结束检测	feof（文件指针）;
	ferror( )	读写文件出错检测	出错类型码存放变量 = ferror（文件指针）;
	clearerr( )	文件出错或结束标志置 0	clearerr（文件指针）;

第 2 部分　补充练习题

第 1，2 章　C语言概述及数据类型与表达式

一、单项选择题

1．C 语言源程序名的后缀是_____。

A．exe　　B．c　　C．obj　　D．cpp

2．以下叙述中错误的是_____。

A．计算机不能直接执行用 C 语言编写的源程序

B．C 程序经 C 编译程序编译后，生成后缀为.obj 的文件是一个二进制文件

C．后缀为.obj 的文件，经连接程序生成后缀为.exe 的文件是一个二进制文件

D．后缀为.obj 和.exe 的二进制文件都可以直接运行

3．可在 C 程序中作为用户标识符的一组是_____。

A．and	B．Date	C．Hi	D．case
_2007	y−m−d	Dr.Tom	Bigl

4．以下选项中，合法的一组 C 语言数值常量是_____。

A．028	B．12.	C．.177	D．0x8A
.5e−3	0xa23	4e1.5	10 000
.0xf	4.5e0	0abc	3.e5

5．以下关于 long、int 和 short 类型数据占用内存大小的叙述正确的是_____。

A．均占 4 个字节

B．根据数据的大小来决定所占内存的字节数

C．由用户自己定义

D．由 C 语言编译系统决定

6．若变量均已正确定义并赋值，则以下合法的 C 语言赋值语句是_____。

A．x＝y＝＝5;　　B．x＝n%2.5;　　C．x＋n＝i;　　D．x＝5＝4+1;

7．以下正确的字符串常量是_____。

A．" \\\"　　B．'abc'　　C．Olympic Games　　D．""

8．已知字符'A'的 ASCII 代码值是 65，字符变量 c1 的值是'A'，c2 的值是'D'。执行语句 printf("%d,%d",c1,c2−2);后输出结果是_____。

A．A, B　　B．A, 68　　C．65, 66　　D．65, 68

9．执行 scanf ("a＝%d,b＝%d",&a,&b);语句，若要使变量 a 和 b 的值分别为 3 和 4，则正确的输入方法为_____。

A．3,4　　B．a:3　b:4　　C．a＝3,b＝4　　D．3　4

10．设变量均已正确定义，若要通过 scanf("%d%c%d%c",&a1,&c1,&a2,&c2);语句为变量

a1 和 a2 赋数值 10 和 20，为变量 c1 和 c2 赋字符 X 和 Y。以下所示的输入形式中正确的是（注：□代表空格字符）_____。

A．10□X□20□Y<回车>　　B．10□X20□Y<回车>

C．10□X<回车>　　D．10X<回车>

20□Y<回车>　　20Y<回车>

11．转换说明符%x 的输出形式是_____。

A．十六进制数　　B．八进制数　　C．十进制数　　D．二进制数

12．若有以下类型说明语句：

```
char        w;
int         x;
float       y;
double      z;
```

则表达式 w*x+z–y 的结果为_____类型。

A．float　　B．double　　C．int　　D．char

13．将 int 型变量 n 转换成 float 型变量的方法是_____。

A．float n　　B．(float)n　　C．float(n)　　D．int n

14．以下不能正确计算代数式$\frac{1}{3}\sin^2\left(\frac{1}{2}\right)$值的 C 语言表达式是_____。

A．1/3*sin(1/2) *sin(1/2)　　B．sin(0.5) *sin(0.5)/3

C．pow(sin(0.5),2)/3　　D．1/3.0*pow(sin(1.0/2),2)

15．C 语言中运算对象必须是整型的运算符是_____。

A．/　　B．%　　C．!　　D．*

16．运行以下程序段时编译出错，其原因是_____。

```
char c1='a',  c2='123';
printf( "%c,%d\n", c1, c2 );
```

A．字符串要用"123"表示

B．'123'只能赋值给字符数组

C．c2 是字符变量，不能用%d 格式输出

D．c2 是字符变量，只能赋以字符常量，不能赋以字符串常量

17．设有定义：int k = 0；以下选项的四个表达式中与其他三个表达式的值不相同的是_____。

A．k++　　B．k+=1　　C．++k　　D．k+1

18．已知大写字母 A 的 ASCII 码是 65，小写字母 a 的 ASCII 码是 97，以下不能将变量 c 中的大写字母转换成对应小写字母的语句是_____。

A．`c = (c-'A')%26+'a';`　　B．`c = c+32;`

C．`c = c-'A'+'a';`　　D．`c = ('A'+c)%26-'a';`

19．有以下程序：

```
#include <stdio.h>
void main()
{   int x, y, z;
    x=y=1;
```

```
    z=x++,y++,++y;
    printf("%d,%d,%d\n", x,y,z);
}
```

程序运行后的输出结果是_____。

A．2,3,3　　B．2,3,2　　C．2,3,1　　D．2,2,1

20．指出下面输入函数正确的是 _____。

A．`scanf("a=b=%d",&a,&b);`　　B．`scanf("a=%d,b=%f",&m,&f);`

C．`scanf("%3c",c);`　　D．`scanf("%5.2f",&f);`

21．有以下程序：

```
#include <stdio.h>
void main()
{   char c1,c2,c3,c4,c5,c6;
    scanf ("%c%c%c%c",&c1,&c2,&c3,&c4);
    c5=getchar();
    c6=getchar();
    putchar(c1);
    putchar(c2);
    printf("%c%c\n",c5,c6);
}
```

程序运行后，若从键盘输入(从第 1 列开始)

```
123<回车>
45678<回车>
```

则输出结果是_____。

A．1267　　B．1256　　C．1278　　D．1245

22．以下程序的功能是：给 r 输入数据后计算半径为 r 的圆面积 s。

```
#include <stdio.h>
void main()
/* Beginning */
{   int r;
    float s;
    scanf("%d",&r);
    s=π*r*r;
    printf("s=%f\n",s);
}
```

程序在编译时出错，出错的原因是_____。

A．注释语句书写位置错误

B．存放圆半径的变量 r 不应该定义为整型

C．输出语句中格式描述符非法

D．计算圆面积的赋值语句中使用了非法变量

23．以下程序的输出结果是_____。

```
#include <stdio.h>
void main()
{   int  m=3,n=4,x;
    x=-m++;
    x=x+8/++n;
```

```
    printf("%d\n",x);
}
```

A．3　　B．5　　C．–1　　D．–2

24．以下程序的输出结果是_____。

```
#include <stdio.h>
void main()
{   int x=4,y=8;
    x+=y;
    y+=x;
    printf("%d %d\n",x,y);
}
```

A．12　20　　B．4　8　　C．12　12　　D．8　4

25．下列程序的执行结果是_____。

```
#include <stdio.h>
void main()
{   int x='c';
    printf ("%c\n",'B'+(x-'b'+1));
}
```

A．D　　B．H　　C．I　　D．J

26．下列程序的执行结果是_____。

```
#include <stdio.h>
void main()
{   int a=5;
    float x=2.14;
    a*=x*('D'-'A');
    printf("%f\n",(float)a);
}
```

A．32.100000　　B．32　　C．32.000000　　D．33.000000

27．下列程序的执行结果是_____。

```
#include <stdio.h>
void main()
{   int  x=0,y=0,z=0;
    z=(x-=x-4),(x=y,y+4);
    printf("%d,%d,%d\n",x,y,z);
}
```

A．4, 0, –8　　B．0, 0, 4　　C．–8, 4, –8　　D．4, 0, 4

28．数字字符 0 的 ASCII 值为 48，以下程序运行后的输出结果是_____。

```
#include <stdio.h>
void main()
{   char a='1',b='2';
    printf("%c,",b++);
    printf("%d\n",b-a);
}
```

A．3, 2　　B．50, 2　　C．2, 2　　D．2, 50

29．有以下程序：

```
#include <stdio.h>
```

```
void main()
{   char a1='M', a2='m';
    printf("%c\n",(a1, a2));
}
```

以下叙述中正确的是_____。

A．程序输出大写字母 M　　B．程序输出小写字母 m

C．格式说明符不足，编译出错　　D．程序运行时产生出错信息

30．有以下程序：

```
#include <stdio.h>
void main()
{   char c1='1',c2='2';
    c1=getchar();
    c2=getchar();
    putchar(c1);
    putchar(c2);
}
```

当运行时输入：a<回车> 后，以下叙述正确的是_____。

A．变量 c1 被赋予字符 a，c2 被赋予回车符

B．程序将等待用户输入第 2 个字符

C．变量 c1 被赋予字符 a，c2 中仍是原有字符 2

D．变量 c1 被赋予字符 a，c2 中将无确定值

二、填空题

1．一个 C 源程序是由若干个函数构成，其中必须有一个是_____函数。

2．两整数相除的结果为_____类型。

3．函数体由_____开始，由符号_____ 结束。函数体的前面是_____部分，其后是_____部分。

4．在 C 语言中整数可用_____进制数、_____进制数和_____进制数三种数制表示。

5．在 C 语言程序中，整型数据占_____字节，用关键字_____定义单精度实型变量，字符型数据占_____字节。

6．执行以下程序后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int a=10;
    a=(3*5,a+4);
    printf("a=%d\n",a);
}
```

7．设有定义：float x=123.4567;则执行以下语句后的输出结果是_____。

```
printf("%f\n",(int)(x*100+0.5)/100.0);
```

8．以下程序运行后的输出结果是_____。

```
#include <stdio.h>
void main()
{   char c;
    int n=100;
```

```
    float f=10;
    double x;
    x=f*=n/=(c=50);
    printf("%d，%f\n",n,x);
}
```

9．已知字母 A 的 ASCII 码为 65。以下程序运行后的输出结果是_____。

```
#include <stdio.h>
void  main()
{   char a, b;
    a='A'+'5'-'3';
    b=a+'6'-'2';
    printf("%d，%c\n",a,b);
}
```

10．有以下程序，当运行时输入：12<回车> 后，输出结果是_____。

```
#include <stdio.h>
void main()
{   char ch1,ch2;
    int n1,n2;
    ch1=getchar();
    ch2=getchar();
    n1=ch1-'0';
    n2=n1*10+(ch2-'0');
    printf("%d\n",n2);
}
```

11．执行以下程序后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int x,y,z;
    x=y=2;
    z=3;
    y=z++-1;
    printf("%d\t%d\t",x,y);
    y=++x-1;
    printf("%d \t%d\n",x,y);
    y=z---1;
    printf("%d\t%d\t",z,y);
    y=--z-1;
    printf("%d\t%d\n",z,y);
}
```

12．执行以下程序后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int x;
    x=-3+4*5-6;
    printf("%d,\t",x);
    x=3+4%5-6;
    printf("%d,\t",x);
    x=(7+6)%5%2;
    printf("%d\n",x);
}
```

13．执行以下程序后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int a,b=97;
    a=-3;
    printf("\ta=%d\tb=\'%c\'\t\"end\"\n",a,b);
}
```

14．以下程序输入 1.2345.6789<CR>，则程序运行结果为 x = 1.230000, y = 45.678900，请填空完成程序。注：<CR>表示回车。

```
#include <stdio.h>
void main()
{   double x, y;
    scanf (_____________);
    printf("x=%lf,y=%lf\n", x, y);
 }
```

15．以下程序的运行结果是_____。

```
#include <stdio.h>
void main()
{   int i=5, j=9;
    float x=2.3, y=4.5;
    printf("%8.2f\n", i%(int)(x+y)*j/2/3+y);
}
```

第 3 章　算法的基本控制结构

一、单项选择题

1．在以下运算符中，优先级最高的运算符是_____。

A．<=　　B．/　　C．!=　　D．&&

2．以下叙述中错误的是_____。

A．C 语句必须以分号结束

B．复合语句在语法上被看做一条语句

C．空语句出现在任何位置都不会影响程序运行

D．赋值表达式末尾加分号就构成赋值语句

3．设有定义：int k=1,m=2;float f=7.0;则以下选项中错误的表达式是_____。

A．k = k >= k　　B．–k++　　C．k%int(f)　　D．k >= f >= m

4．当把以下四个表达式用做 if 语句的控制表达式时，有一个选项与其他三个选项含义不同，这个选项是_____。

A．k%2　　B．k%2 = = 1　　C．(k%2)! = 0　　D．!k%2 = = 1

5．设有定义：int a = 2,b = 3,c = 4;则以下选项中值为 0 的表达式是_____。

A．(!a = = 1)&&(!b = = 0)　　B．(a<b)&&!c||1

C．a&&b　　D．a||(b+b)&&(c–a)

6．有以下程序段

```
int k=0,a=1,b=2,c=3;
k=a<b?b:a; k=k>c?c:k;
```

执行该程序段后，k 的值是_____。

A．3　　B．2　　C．1　　D．0

7．设变量 a, b, c, d 和 y 都已正确定义并赋值。若有以下 if 语句

```
if(a<b)
    if(c==d )  y=0;
else  y=1;
```

该语句所表示的含义是_____。

A．$y=\begin{cases}0 & a<b \text{ 且 } c=d\\ 1 & a\geqslant b\end{cases}$　　B．$y=\begin{cases}0 & a<b \text{ 且 } c=d\\ 1 & a\geqslant b \text{ 且 } c\neq d\end{cases}$

C．$y=\begin{cases}0 & a<b \text{ 且 } c=d\\ 1 & a<b \text{ 且 } c\neq d\end{cases}$　　D．$y=\begin{cases}0 & a<b \text{ 且 } c=d\\ 1 & c\neq d\end{cases}$

8．若整型变量 a, b, c, d 中的值依次为 1, 4, 3, 2，则条件表达式 a<b?a:c<d?c:d 的值为_____。

A．1　　B．2　　C．3　　D．4

9．以下程序运行后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int i=1,j=2,k=3;
    if(i++==1&&(++j==3||k++==3))
    printf("%d %d %d\n",i,j,k);
}
```

A．1 2 3　　B．2 3 4

C．2 2 3　　D．2 3 3

10．以下程序运行后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int a,b,d=25;
    a=d/10%9;b=a&&(-1);
    printf("%d,%d\n",a,b);
}
```

A．6,1　　B．2,1　　C．6,0　　D．2,0

11．以下程序运行后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int a=1,b;
    for(b=1;b<=10;b++)
    {   if(a>=8)  break;
        if(a%2==1)
        {   a+=5;
            continue;
        }
        a-=3;
```

```
    }
    printf("%d\n",b);
}
```

A．3　　B．4　　C．5　　D．6

12．有以下程序段

```
int n,t=1,s=0;
scanf("%d",&n);
do{ s=s+t; t=t-2;  }while (t!=n);
```

为使此程序段不陷入死循环，从键盘输入的数据应该是_____。

A．任意正奇数　　B．任意负偶数

C．任意正偶数　　D．任意负奇数

13．设变量已正确定义，则以下能正确计算 f=n!的程序段是_____。

A．f = 0;
　for(i = 1;i <= n;i++)
　　f∗ = i;

B．f = 1;
　for(i = 1;i<n;i++)
　　f∗ = i;

C．f = 1;
　for(i = n;i>1;i++)
　　f∗= i;

D．f = 1;
　for(i = n;i >= 2;i—)
　　f∗ = i;

14．设 a 为整型变量，不能正确表达数学关系: 10<a<15 的 C 语言表达式是_____。

A．10<a<15　　B．a = = 11||a = = 12||a = = 13||a = = 14

C．a>10&&a<15　　D．!(a <= 10)&&!(a >= 15)

15．逻辑运算符两侧运算对象的数据类型_____。

A．只能是 0 或 1　　B．只能是 0 或非 0 正数

C．只能是整型或字符型数据　　D．可以是任何类型的数据

16．当执行以下程序段时,供选择的答案是_____。

```
 a=-3; do {  a+=a;  } while(!a);
```

A．循环体将执行一次　　B．循环体将执行两次

C．循环体将执行无限次　　D．系统提示有语法错误

17．若定义 int k; 则以下循环语句的循环执行次数是_____。

```
for(k=2;k==0;)  printf("%d", k--);
```

A．无限　　B．0　　C．1　　D．2

18．为了避免嵌套的条件语句 if~else 的二义性，C 语言规定_____。

A．else 与缩排位置相同的 if 配对　　B．else 与其之前最近的 if 配对

C．else 与其之后最近的 if 配对　　D．else 与同一行上的 if 配对

19．以下不正确的描述是_____。

A．用 while 和 do~while 循环时，循环变量初始化的操作应在循环语句前完成

B．while 循环是先判断表达式，后执行循环体语句

C．do~while 和 for 循环均是先执行循环体语句，后判断表达式

D．for，while 和 do~while 循环中的循环体语句均可以由空语句构成

20．执行以下程序后的输出结果是_____。

```
#include <stdio.h>
```

```
void main()
{   int a, b, c;
    a=1;
    b=2;
    c=3;
    a=b--<=a||a+b!=c;
    printf("%d,%d\n",a,b);
}
```

A.1,1　　B. 1,2　　C. 2,1　　D. 2,2

21. 以下程序运行后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int k=5,n=0;
    while(k>0)
    {   switch(k)
        {   default: break;
            case 1: n+=k;
            case 2:
            case 3: n+=k;
        }
        k--;
    }
    printf("%d\n",n);
}
```

A. 0　　B. 4　　C. 6　　D. 7

22. 若 x 和 y 代表整型数，以下表达式中不能正确表示数学关系|x–y|<10 的是_____。

A. abs(x–y)<10　　B. x–y>–10&&x–y<10

C. !(x–y)<–10||!(y–x)>10　　D. (x–y) * (x–y)<100

23. 下列条件语句中，功能与其他语句不同的是_____。

A. `if(a)  printf("%d\n",x); else printf("%d\n",y);`

B. `if(a==0) printf("%d\n",y); else printf("%d\n",x);`

C. `if(a!=0) printf("%d\n",x); else printf("%d\n",y);`

D. `if(a==0) printf("%d\n",x); else printf("%d\n",y);`

24. 以下程序的输出结果是_____。

```
#include <stdio.h>
void main()
{   int i=0,s=0;
    for (;;)
    {   if(i==3||i==5)  continue;
        if(i==6)  break;
        i++;
        s+=i;
    }
    printf("%d\n",s);
}
```

A. 10　　B. 13　　C. 21　　D. 程序是死循环

25. 以下程序的输出结果是_____。

```
#include <stdio.h>
```

```
void main()
{   int a=3,b=4,c=5,d=2;
    if(a>b)
        if(b>c)
          printf("%d",d+++1);
        else
          printf("%d",++d+1);
    printf("%d\n",d);
}
```

A．2　　B．3　　C．43　　D．44

26．若有定义：float x = 1.5; int a = 1, b = 2, c = 3; 则正确的 switch 语句是_____。

A．
```
switch(x)
{ case 1.0: printf("*\n");
  case 2.0: printf("**\n"); }
```

B．
```
switch((int)x);
{ case 1: printf("*\n");
  case 2: printf("**\n"); }
```

C．
```
switch(a+b)
{ case 1: printf("*\n");
  case 2: printf("**\n");
  default: printf("***\n"); }
```

D．
```
switch(a+b)
{ case 1: printf("*\n");
  default: printf("***\n");
  case c: printf("**\n"); }
```

27．在以下给出的表达式中，与 while(E)中的(E)不等价的表达式是_____。

A．(!E==0)　　B．(E>0‖E<0)　　C．(E==0)　　D．(E!=0)

28．要求通过 while 循环不断读入字符,当读入字母 N 时结束循环。若变量已正确定义，以下正确的程序段是_____。

A．`while((ch=getchar( ))!='N') printf("%c",ch);`

B．`while(ch=getchar( )!='N')  printf("%c",ch);`

C．`while(ch=getchar( )=='N')  printf("%c",ch);`

D．`while((ch=getchar( ))=='N') printf("%c",ch);`

29．有以下程序：

```
#include <stdio.h>
void main()
{   int x=6;
    while(x--);
    printf("x=%d\n",x);
}
```

运行程序的输出结果是_____。

A．x = 0　　B．x = –1

C．x = 1　　D．while 构成无限循环

30．有以下程序：

```
#include <stdio.h>
void main()
{   int a=0,b=0,c=0,d=0;
    if(a=1)  b=1;c=2;
    else d=3;
    printf("%d,%d,%d,%d\n",a,b,c,d);
}
```

运行程序的输出结果是_____。

A．0,1,2,0　　B．0,0,0,3

C．1,1,2,0　　D．编译有错

二、填空题

1．表示条件：10<z<100 或 z<0 的 C 语言表达式是_____。

2．若已知 i=10，j=20，则表达式!i<j 的值为_____。

3．当 x=6，y=5，z=4 时，表达式 a=x>y>z 的值是_____。

4．阅读下列程序：

```
#include <stdio.h>
void main()
{   char  c;
    while((c=getchar())!='\n')
        {  if (c>='A' && c<='Z')  c=c+32;
           else if(c>='a' && c<='z')  c=c-32;
           printf("%c",c);
        }
    printf("\n");
}
```

执行时，如果从键盘上输入：DEFabc<回车>，则运行结果是_____。

5．输入：742<回车>，以下程序的运行结果是_____。

```
#include <stdio.h>
void main()
{   int c;
    while((c=getchar())!='\n')
    {   switch(c-'2')
        {   case 0:
            case 1: putchar(c+4);
            case 2: putchar(c+4);break;
            case 3: putchar(c+3);
            default: putchar(c+2);break;
        }
    }
    printf("\n");
}
```

6．执行以下程序后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int i, j, x=0;
    for(i=0;i<2;i++)
        {   x++;
            for(j=0;j<3;j++)
                if(j%2)  break;
            x++;
        }
    printf("x=%d\n", x);
}
```

7．以下程序的功能是：输出 a, b, c 三个变量中的最小值，请填空。

```
#include <stdio.h>
void main()
{   int a,b,c,t1,t2;
    scanf("%d%d%d",&a,&b,&c);
    t1=a<b?_______;
    t2=c<t1?_______;
    printf("%d\n",t2);
}
```

8．有以下程序段，且变量已正确定义和赋值：

```
for(s=1.0,k=1;k<=n;k++)
    s=s+1.0/(k*(k+1));
printf("s= %f\n", s);
```

请填空，使下面程序段的功能与之完全相同。

```
s=1.0;k=1;
while(_______)
{   s=s+1.0/(k*(k+1));
    _______;
}
printf("s=%f\n",s);
```

9．以下程序的输出结果是_____。

```
#include <stdio.h>
void main()
{   int i;
    for( i=1;i<6;i++)
    {   if (i%2)  printf("#");
        else   continue;
        printf("*");
    }
    printf("\n");
}
```

10．以下程序的功能是：一球从 100 米高度垂直自由落下，每次落地后反弹回上次高度的一半，再落下……。以下程序计算该球第 10 次反弹高度（米），以及球第 10 次落地时经过的路程（米）。

```
#include <stdio.h>
void main()
{   float sn=100.0,hn=____;
    int n;
    for(n=2;n<=10;n++)
    {  _____;       /* sn 为球第 n 次落地时共经过的路程（米），n>=2 */
       hn/=2;       /* hn 为球第 n 次反弹的高度（米），n>=2 */
    }
    printf("球第 10 次落地时共经过%f 米\n",sn);
    printf("球第 10 次反弹的高度为%f 米\n",hn);
}
```

11．求 s = 1+12+123 + 1234+12345。程序如下，请填空。

```
#include <stdio.h>
void main()
```

```
{   int t=0,i,s=0;
    for(i=1;i<=5;i++)
    {   t=i+_______;
        s=_______;  }
    printf("s=%d",s);
}
```

12．以下程序的功能是输出如下形式的方阵，请填空。

```
13  14  15  16
 9  10  11  12
 5   6   7   8
 1   2   3   4
#include <stdio.h>
void main()
{   int i,j,x;
    for(j=4;_______;j--)
    {   for(i=1;i<=4;i++)
        {   x=(j-1)*4 +_______;
            printf("%4d",x);
        }
        printf("\n");
    }
}
```

13．以下程序用于判断 a, b, c 能否构成三角形，若能，则输出 YES，否则输出 NO。当给 a, b, c 输入三角形三条边长时，确定 a, b, c 能构成三角形的条件是需同时满足三个条件：a+b>c，a+c>b，b+c>a。请填空。

```
#include <stdio.h>
void main()
{   float a,b,c;
    scanf("%f %f %f",&a,&b,&c);
    if(_______)
      printf("YES\n");          /* a,b,c 能构成三角形 */
    else  printf("NO\n");        /* a,b,c 不能构成三角形 */
  }
```

14．以下程序的功能是：输出 100 以内（不含 100）能被 3 整除且个位数为 6 的所有整数，请填空。

```
#include <stdio.h>
void main()
{   int i,j;
    for(i=0;_______;i++)
    {   j=i*10+6;
        if(_______)continue;
        printf("%d\n",j);
    }
}
```

15．以下程序的功能是输入任意整数给 n 后，输出 n 行由大写字母 A 开始构成的三角形字符阵列图形。例如，输入整数 5 时（注意：n 不得大于 10），程序运行情况如下，请填空完成该程序。

```
A B C D E
F G H I
```

```
J K L
M N
O
#include <stdio.h>
void main()
{   int i,j,n;
    char ch='A';
    scanf("%d",&n);
    if(n<11)
    {   for(i=1;i<=n;i++)
        {   for(j=1;j<=n-i+1;j++)
            {   printf("%2c",ch);
                _______;
            }
            _______;
        }
    }
    else printf("n is too large!\n");
    printf("\n");
}
```

第 4 章　函　　数

一、单项选择题

1．以下叙述中正确的是_____。

A．C 语言程序将从源程序中第一个函数开始执行

B．可以在程序中由用户指定任意一个函数作为主函数，程序将从此开始执行

C．C 语言规定必须用 main 作为主函数名，程序将从此开始执行，在此结束

D．main 可作为用户标识符，用以命名任意一个函数作为主函数

2．以下叙述中错误的是_____。

A．C 程序必须由一个或一个以上的函数组成

B．函数调用可以作为一个独立的语句存在

C．若函数有返回值，必须通过 return 语句返回

D．函数形参的值也可以传回给对应的实参

3．以下叙述中正确的是_____。

A．C 程序中注释部分可以出现在程序中任意合适的地方

B．花括号"{"和"}"只能作为函数体的定界符

C．构成 C 程序的基本单位是函数，所有函数名都可以由用户命名

D．分号是 C 语句之间的分隔符，不是语句的一部分

4．若已定义的函数有返回值，则在以下关于该函数调用的叙述中错误的是_____。

A．函数调用可以作为独立的语句存在

B．函数调用可以作为一个函数的实参

C．函数调用可以出现在表达式中

D．函数调用可以作为一个函数的形参

5．设函数 fun 的定义形式为 void fun(char ch, float x) { … }，则以下对函数 fun 的调用语句中，正确的是_____。

A．fun("abc",3.0);　　B．t=fun('D',16.5);

C．fun('65',2.8);　　D．fun(32,32);

6．调用函数的实参与被调用的形参应该有如下关系_____。

A．只要求实参和形参个数相等

B．只要求实参和形参顺序相同

C．只要求实参和形参数据类型相同

D．上述三点均需具备

7．在 C 语言中，以下不正确的说法是_____。

A．实参可以是常量、变量或表达式

B．形参可以是常量、变量或表达式

C．实参可以为任意类型

D．形参应与其对应的实参类型一致

8．以下关于说法正确的是_____。

A．实参和与其对应的形参各自占用独立的存储单元

B．实参与其对应的形参共占用一个存储单元

C．实参和与其对应的形参同名时才共同占用存储单元

D．形参是虚拟的，不占用存储单元

9．在函数调用过程中，如果函数 funA 调用了函数 funB，函数 funB 又调用了函数 funA，则_____。

A．称为函数的直接递归调用　　B．称为函数的间接递归调用

C．称为函数的循环调用　　D．C 语言中不允许这样的递归调用

10．有以下程序：

```
#include <stdio.h>
fun(int x,int y)
{   return(x+y);  }
void main()
{   int a=1,b=2,c=3,sum;
    sum=fun( (a++,b++,a+b),c++);
    printf("%d\n",sum);
}
```

程序运行后的输出结果是_____。

A．6　　B．7　　C．8　　D．9

11．有以下程序：

```
#include <stdio.h>
int fun1(double a)
{   return a*=a;   }
int fun2(double x,double y)
{   double a=0,b=0;
       a=fun1(x);b=fun1(y);return(int)(a+b);
}
```

```
void main()
{   double w;
    w=fun2(1.1,2.0);
}
```

程序执行后变量 w 中的值是_____。

A．5.21　　B．5　　C．5.000000　　D．0.0

12．有以下程序：

```
#include <stdio.h>
int f1(int x,int y){  return x>y?x:y;  }
int f2(int x,int y){  return x>y?y:x;  }
void main()
{   int a=4,b=3,c=5,d=2,e,f,g;
    e=f2(f1(a,b),f1(c,d));f=f1(f2(a,b),f2(c,d));
    g=a+b+c+d-e-f;
    printf("%d,%d,%d\n",e,f,g);
}
```

程序运行后的输出结果是_____。

A．4,3,7　　B．3,4,7　　C．5,2,7　　D．2,5,7

13．有以下程序：

```
#include <stdio.h>
int sum(int n)
{   int p=1,s=0,i;
    for(i=1;i<=n;i++)  s+=(p*=i);
    return s;
}
void main()
{   printf("sum(5)=%d\n", sum(5));
}
```

程序运行后的输出结果是_____。

A．sum(5)=151　　B．sum(5)=152

C．sum(5)=153　　D．sum(5)=155

14．有以下程序：

```
#include <stdio.h>
fun(int x)
{   int p;
    if(x==0||x==1)
      return(3);
    p=x-fun(x-2);
    return p;
}
void main()
{   printf("%d\n", fun(7));  }
```

程序运行后的输出结果是_____。

A．7　　B．3　　C．2　　D．0

15．有以下程序：

```
#include <stdio.h>
int fun(int n)
```

```
{   if(n==1)  return 1;
    else  return(n+fun(n-1));  }
void main()
{   int x;
    scanf("%d",&x);
    x=fun(x);
    printf("%d\n",x);
}
```

执行程序时，给变量 x 输入 10，程序的输出结果是_____。

A．55　　B．54　　C．65　　D．45

16．以下程序的输出结果是_____。

```
#include <stdio.h>
void f(int v, int w)
{   int t;
    t=v;v=w;w=t;
}
void main()
{   int x=1,y=3,z=2;
    if(x>y)  f(x,y);
    else if(y>z)  f(y,z);
    else    f(x,z);
    printf("%d,%d,%d\n",x,y,z);
}
```

A．1,2,3　　B．3,1,2　　C．1,3,2　　D．2,3,1

17．有以下程序：

```
#include <stdio.h>
fun(int a, int b)
{   if(a>b) return(a);
    else    return(b);  }
void main()
{   int x=3, y=8, z=6, r;
    r=fun(fun(x,y), 2*z);
    printf("%d\n", r);
}
```

运行程序的输出结果是_____。

A．3　　B．6　　C．8　　D．12

18．下列程序的输出结果是_____。

```
#include <stdio.h>
fun(int a, int b, int c)
{
    c=a*b;
}
void main()
{
    int c;
    fun(2, 3, c);
    printf("%d\n", c);
}
```

A．0　　B．6　　C．1　　D．无法确定

19．C 语言规定：在一个源程序中，main 函数的位置_____。

A．必须在最开始　　B．可以任意

C．必须在系统调用的库函数的后面　　D．必须在最后

20．若各选项中所用变量已正确定义，函数 fun 中通过 return 语句返回一个函数值，以下选项中错误的程序是_____。

A．
```
void main()
{……x=fun(2，10);……}
float fun(int a，int b){……}
```

B．
```
float fun(int a，int b){……}
void main( )
{……x=fun(i，j);……}
```

C．
```
float fun(int，int);
void main()
{……x=fun(2，10);……}
float fun(int a，int b){……}
```

D．
```
void main()
{ float fun(int a，int b);
……x=fun(i，j);……}
float fun(int a，int b){……}
```

二、填空题

1．若自定义函数不要求返回一个值，则应在函数说明时加一个类型说明符_____。

2．执行以下程序后的输出结果是_____。

```
#include <stdio.h>
int f(int x,int y)
{   return(y-x)*x;  }
void main()
{   int a=3,b=4,c=5,d;
    d=f(f(3,4),f(3,5));
    printf("%d\n",d);
}
```

3．下面的函数是一个求非负整数阶乘的递归函数。请填空。

```
fac(int n)
{   if(n==0)_____;
    else return _____;
}
```

4．以下程序是用来从 10 到 99 数字中挑出能被 3 整除且至少有一位是 5 的两位数，并打印出所有这样的数及其个数。请填空。

```
#include <stdio.h>
void main()
{   int k,m=0;
    for(k=10;k<=99;k++)
        {  m+=sub(k);   }
    printf("个数为%d\n",m);
}
sub(int k)
{   int a1,a2,n=0;
    a2=_____;
    a1=k/10;
    if(k%3==0&&(_____))
    {   printf("%d",k);
        return 1;
    }
    else return 0;
}
```

5．以下函数 abc 的功能是_____。

调用该函数后，从键盘输入 4，则整个函数的运行结果为_____。

```
void abc()
{   int choice;
    do
    {   printf("*** 主菜单* * *\n");
        printf("1.输入       2.排序\n");
        printf("3.输出       4.退出\n");
        printf("请选择 1-4");
        scanf("%d",&choice);
        switch(choice)
        {   case 1: sr();  break;  /* sr()为自定义输入函数*/
            case 2: sort();break;  /* sort()为自定义排序函数*/
            case 3: sc();  break;  /* sc()为自定义输出函数*/
            case 4: exit(0);  }
        } while (1);
    }
```

6．以下 isprime 函数的功能是判断形参 a 是否为素数，若是素数，则函数返回 1，否则返回 0。请填空。

```
int isprime(int a)
{   int i;
    for(i=2;i<=a/2;i++)
    if(a%i==0)_______;
    _______;
}
```

7．以下函数 utoh 将十进制正整数 n 转换成无符号十六进制数并输出，请填空。

```
utoh(unsigned n)
{   int h;
    char ch;
    h=n%16;
    ch=h<=9?h+'0':h+'A'-10;
    if(n/16>0)   utoh(_______);
    printf("%c",ch);
}
```

8．以下程序中，函数 fun 的功能是计算 x^2-2x+6，主函数中将调用函数计算:

$$y1=(x+8)^2-2(x+8)+6$$

$$y2=\sin^2(x)-2\sin(x)+6$$

请填空。

```
#include <stdio.h>
#include <math.h>
double fun(double x)
{   return (x*x-2*x+6 );  }
 void main()
{   double x,y1,y2;
    printf ("Enter x:");
    scanf ("%lf",&x);
    y1=fun(_______);
    y2=fun(_______);
```

```
    printf ("y1=%lf ,y2=%lf\n",y1,y2);
}
```

9．求以下分数序列的前 n 项之和。请填空。

$$\frac{2}{1},\frac{3}{2},\frac{5}{3},\frac{8}{5},\frac{13}{8},\frac{21}{13},\cdots$$

```
#include <stdio.h>
_______ fun(int n)
{   int a=2,b=1,c,k;
    double s=0.0;
    for(k=1;k<=n;k++)
    {   s+=(double)a/b;
        c=a;
        a=a+b;
        b=c;
    }
    _______;
}
void main()
{   int n=5;
    printf("%lf\n",fun(n));
}
```

10．以下程序通过函数 sunFun 求 $\sum_{x=0}^{10} f(x)$，其中 $f(x)=x^2+1$，由 F 函数实现。请填空。

```
#include <stdio.h>
void main()
{   printf("The sum=%d\n",SunFun(10));  }
SunFun(int n)
{   int x,s=0;
    for(x=0;x<=n;x++) s+=F(__________);
    return s;
}
F(int x)
{  return _______;  }
```

第 5 章　数　　组

一、单项选择题

1．语句 char str[20];说明 str 是一个字符串，最多能表示_____。

A．20 个字符　　B．19 个字符　　C．18 个字符　　D．21 个字符

2．在以下关于字符串的叙述中，正确的是_____。

A．C 语言中有字符串类型的常量和变量

B．两个字符串中的字符个数相同时才能进行字符串大小的比较

C．可以用关系运算符对字符串的大小进行比较

D．空串一定比空格打头的字符串小

3．有以下程序：

```
#include  <stdio.h>
```

```
#define MAX 10
void main ()
{   int i,sum,a[ ]={1,2,3,4,5,6,7,8,9,10};
    sum=1;
    for(i=0;i<MAX;i++)
      sum-=a[i];
    printf("SUM=%d",sum);
}
```

程序运行结果是_____。

A．SUM = 55　　B．SUM = –54　　C．SUM = –55　　D．SUM = 54

4．有以下程序：

```
#include <stdio.h>
void main()
{   char a[40],b[40];
    int i,j;
    printf("Enter the string:");
    scanf("%s",a);
    i=j=0;
    while(a[i]!='\0')
    {   if(!(a[i]>= '0'&&a[i]<= ' 9'))
        {   b[j]=a[i]; j++;
        }
        ++i;
    }
    b[j] = '\0';
    printf("%s",b);
}
```

程序运行后输出的结果是_____。

A．把键盘输入的字符串显示在屏幕上

B．把键盘输入的字符串中的数字字符删掉，然后显示该字符串

C．把键盘输入的字符串中的字符 0 和 9 删掉，然后显示该字符串

D．只保留由键盘输入的字符串中的字母数字，然后显示该字符串

5．有以下程序：

```
#include <stdio.h>
void main()
{   char a[80];
    int i;
    printf("Enter the string:");
    scanf("%s",a);
    i=0;
    while(a[i]!='\0')
    {   if(a[i]>='A'&&a[i]<='Z')
           a[i]=a[i]-'A'+'a';
        i++;
    }
    printf("%s",a);
}
```

程序运行后输出的结果是_____。

A．把键盘输入的字符串中的大写字母变换成小写字母，然后显示变换后的字符串

B．把键盘输入的字符串中的数字字符删除，然后显示该字符串

C．把键盘输入的字符串中的小写字母变换成大写字母，然后显示变换后的字符串

D．把键盘输入的字符串原封不动地显示在屏幕上

6．当用户要求输入的字符串中含有空格时，应使用的输入函数是_____。

A．scanf()　　B．getchar()　　C．gets()　　D．getc()

7．以下能对二维数组 a 进行正确初始化的语句是_____。

A．`int a[2][]={{1,0,1},{5,2,3}};`

B．`int a[][3]={{1,2,3}，{4,5,6}};`

C．`int a[2][4]={{1,2,3},{4,5},{6}};`

D．`int a[][3]={{1,0,0},{},{1,1}};`

8．下面程序段的运行结果是_____。

```
char s[6];s="abcd";printf("\"%s\"\n", s);
```

A．Abcd　　B．"abcd"　　C．\"abcd\"　　D．编译出错

9．下列程序运行后输出的结果为_____。

```
#include <stdio.h>
void main()
{   char ch[7]={"12ab56"};
    int  i, s=0;
    for(i=0;ch[i]>='0'&&ch[i]<='9';i+=2)
      s=10*s+ch[i]-'0';
    printf ("%d\n",s);
}
```

A．1
2
5
6

B．12

C．1256

D．1

10．下列程序运行后输出的结果为_____。

```
#include <stdio.h>
void main()
{   int a[6][6],i,j;
    for(i=1;i<6;i++)
    for(j=1;j<6;j++)
      a[i][j]=(i/j)* (j/i);
    for(i=1;i<6;i++)
    {   for(j=1;j<6;j++)
          printf("%2d",a[i][j]);
        printf("\n");
    }
}
```

A．
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1

B．
0 0 0 0 1
0 0 0 1 0
0 0 1 0 0
0 1 0 0 0
1 0 0 0 0

C．
1 0 0 0 0
0 1 0 0 0
0 0 1 0 0
0 0 0 1 0
0 0 0 0 1

D．
1 0 0 0 1
0 1 0 1 0
0 0 1 0 0
0 1 0 1 0
1 0 0 0 1

11．以下程序中，函数 sort 的功能是对 a 所指数组中的数据进行由大到小的排序。

```
#include <stdio.h>
void sort(int a[],int n)
{   int  i,j,t;
    for(i=0;i<n-1;i++)
    for(j=i+1;j<n;j++)
    if(a[i]<a[j])
      { t=a[i];a[i]=a[j];a[j]=t;  }
}
void main()
{   int aa[10]={1,2,3,4,5,6,7,8,9,10},i;
    sort(&aa[3],5);
    for(i=0;i<10;i++)
        printf("%d,",aa[i]);
    printf("\n");
}
```

程序运行后的输出结果是_____。

A．1,2,3,4,5,6,7,8,9,10,　　B．10,9,8,7,6,5,4,3,2,1,

C．1,2,3,8,7,6,5,4,9,10,　　D．1,2,10,9,8,7,6,5,4,3,

12．有以下程序：

```
#include <stdio.h>
#include <string.h>
void main()
{   char a[]={'a','b','c','d','e','f','g','h','\0'};
    int i,j;
    i=sizeof(a);
    j=strlen(a);
    printf("%d,%d\n",i,j);
}
```

程序运行后的输出结果是_____。

A．9,9　　B．8,9　　C．1,8　　D．9,8

13．以下程序中函数 reverse 的功能是将 a 所指数组中的内容进行逆置。

```
#include <stdio.h>
void reverse(int  a[],int  n)
{   int  i,t;
    for(i=0;i<n/2;i++)
    {   t=a[i];a[i]=a[n-1-i]; a[n-1-i]=t;   }
}
void main()
{   int  b[10]={1,2,3,4,5,6,7,8,9,10}; int i,s=0;
    reverse(b,8);
    for(i=6;i<10;i++)  s+=b[i];
    printf("%d\n",s);
}
```

程序运行后的输出结果是 _____。

A．22　　B．10　　C．34　　D．30

14．有以下程序：

```
#include <stdio.h>
```

```
void main()
{   int  aa[4][4]={{1,2,3,4},{5,6,7,8},{3,9,10,2},{4,2,9,6}};
    int  i,s=0;
    for(i=0;i<4;i++)
      s+=aa[i][1];
    printf("%d\n",s);
}
```

程序运行后的输出结果是 _____。

A．11　　B．19　　C．13　　D．5

15. 若有定义语句：int a[3][6]; 则按在内存中的存放顺序，a 数组的第 10 个元素是_____。

A．a[0][4]　　B．a[1][3]　　C．a[0][3]　　D．a[1][4]

16. 以下程序中函数 f 的功能是将 n 个字符串按由大到小的顺序进行排序。

```
#include <stdio.h>
#include <string.h>
void f(char  p[][10],int  n)
{   char t[20];int i,j;
    for(i=0;i<n-1;i++)
    for(j=i+1;j<n;j++)
        if(strcmp(p[i],p[j])<0)
        {   strcpy(t,p[i]);strcpy(p[i],p[j]);strcpy(p[j],t);  }
}
void main()
{   char  p[][10]={"abc","aabdfg","abbd","dcdbe","cd"};
    f(p,5);
    printf("%d\n",strlen(p[0]));
}
```

程序运行后的输出结果是 _____。

A．6　　B．4　　C．5　　D．3

17. 有以下程序：

```
void swap1(int c[])
{   int t;
    t=c[0];c[0]=c[1];c[1]=t;
}
void swap2(int c0,int c1)
{   int t;
    t=c0;c0=c1;c1=t;
}
#include <stdio.h>
void main()
{   int a[2]={3,5},b[2]={3,5};
    swap1(a);  swap2(b[0],b[1]);
    printf("%d  %d  %d  %d\n",a[0],a[1],b[0],b[1]);
}
```

其输出结果是_____。

A．5353　　B．5335　　C．3535　　D．3553

18. 有以下程序：

```
#include <stdio.h>
int f(int b[][4])
```

```
{   int i,j,s=0;
    for(j=0;j<4;j++)
    {   i=j;
        if(i>2)  i=3-j;
        s+=b[i][j];
    }
    return s;
}
void main()
{   int a[4][4]={{1,2,3,4},{0,2,4,5},{3,6,9,12},{3,2,1,0}};
    printf("%d\n",f(a));
}
```

执行后的输出结果是_____。

A．12　　B．11　　C．18　　D．16

19．定义如下变量和数组：

```
int i;
int x[3][3]={1,2,3,4,5,6,7,8,9};
```

则下面语句的输出结果是_____。

```
for (i=0;i<3;i++) printf("%d,",x[i][2-i]);
```

A．1,5,9,　　B．1,4,7,　　C．3,5,7,　　D．3,6,9,

20．有以下程序：

```
#include <stdio.h>
void sort(int a[],int n)
{   int i,j,t;
    for(i=0;i<n-1;i+=2)
    for(j=i+2;j<n;j+=2)
        if(a[i]<a[j])
            {   t=a[i];a[i]=a[j];a[j]=t;  }
}
void main()
{   int aa[10]={1,2,3,4,5,6,7,8,9,10},i;
    sort(aa,10);
    for(i=0;i<10;i++) printf("%d,",aa[i]);
    printf("\n");
}
```

其输出结果是_____。

A．1,2,3,4,5,6,7,8,9,10,　　B．10,9,8,7,6,5,4,3,2,1,

C．9,2,7,4,5,6,3,8,1,10,　　D．1,10,3,8,5,6,7,4,9,2,

21．有以下程序段：

```
int j; float y; char name[50];
scanf("%2d%f%s",&j,&y,name);
```

当执行上述程序段，从键盘上输入 55566　7777abc 后，y 的值为_____。

A．55566.0　　B．566.0　　C．7777.0　　D．566777.0

22．有以下程序：

```
#include <stdio.h>
void f(int a[],int i,int j)
```

```
{   int t;
    if(i<j)
        {   t=a[i]; a[i]=a[j];a[j]=t;
            f(a,i+1,j-1);
        }
}
void main()
{   int i,aa[5]={1,2,3,4,5};
    f(aa,0,4);
    for(i=0;i<5;i++)
         printf("%d,",aa[i]);
    printf("\n");
}
```

执行后输出结果是_____。

A．5,4,3,2,1,　　B．5,2,3,4,1,　　C．1,2,3,4,5,　　D．1,2,3,4,5,

23．有以下程序：

```
#include <stdio.h>
void main()
{   int p[8]={11,12,13,14,15,16,17,18},i=0,j=0;
    while(i++<7)
      if(p[i]%2)
        j+=p[i];
    printf("%d\n",j);
}
```

程序运行后的输出结果是_____。

A．42　　B．45　　C．56　　D．60

24．有以下程序：

```
#include <stdio.h>
#include <string.h>
void main()
{   char a[7]="a0\0a0\0";
    int i,j;
    i=sizeof(a);
    j=strlen(a);
    printf("%d %d\n",i,j);
}
```

程序运行后的输出结果是_____。

A．2 2　　B．7 6　　C．7 2　　D．6 2

25．以下能正确定义一维数组的选项是_____。

A．`int a[5] = {0,1,2,3,4,5};`　　B．`char a[] = {0,1,2,3,4,5};`

C．`char a = {'A','B','C'};`　　D．`int a[5] = "0123";`

26．已有定义：char a[] = "xyz",b[] = {'x','y','z'};，以下叙述中正确的是_____。

A．数组 a 和 b 的长度相同　　B．a 数组长度小于 b 数组长度

C．a 数组长度大于 b 数组长度　　D．上述说法都不对

27．以下叙述中错误的是_____。

A．对于 double 类型数组，不可以直接用数组名对数组进行整体输入或输出

B．数组名代表的是数组所占存储区的首地址，其值不可改变

C．当程序执行中，数组元素的下标超出所定义的下标范围时，系统将给出“下标越界”的出错信息

D．可以通过赋初值的方式确定数组元素的个数

28．有以下程序：

```
#include <stdio.h>
#define N 20
fun(int a[],int n,int m)
{   int i;
    for(i=m;i>=n;i--)a[i+1]=a[i];
}
void main()
{   int i,a[N]={1,2,3,4,5,6,7,8,9,10};
    fun(a,2,8);
    for(i=0;i<5;i++)
    printf("%d",a[i]);
}
```

程序运行后的输出结果是_____。

A．10234　　B．12344　　C．12334　　D．12234

29．不能把字符串：Hello! 赋给数组 b 的语句是_____。

A．char b[10]={'H', 'e', 'l', 'l', 'o', '!','\0'};　　B．char b[10]; b="Hello!";

C．char b[10]; strcpy(b, "Hello!");　　D．char b[10]= "Hello!";

30．有以下程序：

```
#include <stdio.h>
void main()
{   int num[4][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12},{13,14,15,16}},i,j;
    for(i=0;i<4;i++)
      { for(j=0;j<=i;j++) printf("%4c",' ');
        for(j=_______;j<4;j++) printf("%4d",num[i][j]);
        printf("\n");
      }
}
```

若要按以下形式输出数组右上半三角

```
  1     2     3     4
        6     7     8
              11    12
                    16
```

则在程序下划线处应填入的是_____。

A．i－1　　B．i　　C．i+1　　D．4－i

二、填空题

1．设有 n 个人围坐一圈并按顺时针方向 1～n 编号，从第 s 个人开始进行 1 到 m 的报数，报数报到第 m 个人时，此人出圈。再从他的下一个人重新开始 1 到 m 的报数，如此下去，直到所有的人都出圈为止。现要求按此圈排序，每 10 人为一组，给出这 n 个人的排序表。

程序中解此问题的方法是：

① 将 1~n 的序号存入一维数组 p;

② 若第 i 个人报数后出圈，则将 p[i]置于数组的倒数第 i 个位置，而原 i+1 至倒数 i 个元素依次向前移动了一个位置;

③ 重复第②步，直至圈中只剩下 p[1]为止。

```
#include <stdio.h>
#define N 10
#define S 1
void main()
{   int i, j,m,s1,w,p[N+1];
    m=4;s1=S;
    for(i=1;i<=N;i++)  _______
    for(i=N;i>=2;i--)
    {   s1 = _______
        if(s1==0)   _______
        w = p[s1];
        for(j=s1;_______;j++)
            p[j]=p[j+1];
        p[i] = w;
    }
    printf("Sequence coming out from the queue is :\n");
    for(i=N;i>=1;i--)
    {   printf("%4d",p[i]);
        if(_______ %10==0)
            printf("\n");
    }
}
```

2．fun 函数的功能是：首先对 a 所指的 N 行 N 列的矩阵，找出各行中最大的数，再求这 N 个最大值中最小的那个数作为函数值返回。请填空。

```
#define N 100
int fun(int(*a)[N])
{   int  row,col,max,min;
    for(row=0;row<N;row++)
    {   for(max=a[row][0],col=1;col<N;col++)
        if(_____)max=a[row][col];
        if(r ow==0)min=max;
        else  if(____) min=max;
    }
    return min;
}
```

3．下面 rotate 函数的功能是：将 n 行 n 列的矩阵 A 转置，请填空。

```
#define  N   4
void  rotate(int a[][N])
{   int i,j,t;
    for(i=0;i<N;i++)
    for(j=0;___;j++)
    {   t=a[i][j];
        _______;
        a[j][i]=t;
    }
}
```

4. 下面程序的运行结果是_____。

```
#include <stdio.h>
int  f( int a[], int  n)
{   if(n>1)
      return a[0]+f(&a[1],n-1);
    else
      return a[0];
}
void main ()
{   int aa[3]={1,2,3},s;
    s=f(&aa[0],3);
    printf("%d\n",s);
}
```

5. 以下程序运行后的输出结果是_____。

```
#include  <stdio.h>
#include  <string.h>
void main()
{   char ch[]="abc",x[3][4]; int i;
    for(i=0;i<3;i++) strcpy(x[i],ch);
      for(i=0;i<3;i++) printf("%s",&x[i][i]);
    printf("\n");
  }
```

6. 以下程序中，函数 f 的功能是在数组 x 的 n 个数（假定 n 个数互不相同）中找出最大最小数，将其中最小的数与第一个数对换，把最大的数与最后一个数对换。请填空。

```
#include <stdio.h>
void f(int x[],int n)
{   int p0,p1,i,j,t,m;
    i=j=x[0]; p0=p1=0;
    for(m=0;m<n;m++)
    {   if(x[m]>i)
        {   i=x[m]; p0=m;  }
        else if(x[m]<j)
        {   j=x[m]; p1=m;  }
    }
    t=x[p0]; x[p0]=x[n-1]; x[n-1]=t;
    t=x[p1];x[p1]= _____; _____=t;
}
void main()
{   int a[10],u;
    for(u=0;u<10;u++)
        scanf("%d",&a[u]);
    f(a,10);
    for(u=0;u<10;u++)
        printf("%3d",a[u]);
    printf("\n");
}
```

7. 执行以下程序的输出结果是_____。

```
#include <stdio.h>
void main()
{   int i,n[4]={1};
```

```
    for(i=1;i<=3;i++)
    {   n[i]=n[i-1]*2+1;
        printf("%d  ",n[i]);
    }
}
```

8．以下程序的功能是：统计从终端输入字符中每个大写字母的个数，用#号作为输入结束标志。请填空。

```
#include <stdio.h>
#include <ctype.h>
void main ()
{   int num[26],i; char c;
    for (i=0;i<26;i++) num[i]=0;
    while ( _____!= '#')        /* 统计从终端输入的大写字母个数*/
      if (isupper(c)) num[c-65]_____;
    for (i=0;i<26;i++)          /* 输出大写字母和该字母的个数*/
      if (num[i])printf("%c;%d\n",i_____,num[i]);
}
```

9．以下程序的功能是：对从键盘上输入的两个字符串进行比较，然后输出两个字符串中第一个不相同字符的 ASCII 码之差。例如，输入的两个字符串分别为 abcdefg 和 abceef，则输出为–1。请填空。

```
#include <stdio.h>
void main()
{   char str1[100],str2[100],c;
    int i,s;
    printf("\n Input string 1:\n");gets(str1);
    printf("\n Input string 2:\n");gets(str2);
    i=_____;
    while((str1[i]==str2[i])&&(str1[i]!=_____))
        i++;
    s=_____;
    printf("%d\n",s);
}
```

10．以下程序的功能是：从键盘上输入若干个学生的成绩，统计计算出平均成绩，并输出低于平均分的学生成绩，用输入负数结束输入。请填空。

```
#include <stdio.h>
void main()
{   float x[1000], sum=0.0, ave, a;
    int n=0, i;
    printf ("Enter  mark : \n") ;
    scanf("%f", &a);
    while (a>=0.0 && n<1000)
    {   sum+= _______ ;
        x[n]= ______ ;
        n++;
        scanf("%f", &a);
    }
    ave= ______ ;
    printf ("Output : \n");
    printf ("ave = %f\n", ave);
```

```
    for (i=0; i<n; i++)
        if ( ______ )  printf("%f\n", x[i]);
}
```

11. 有以下程序：

```
#include <stdio.h>
#define N 8
void main()
{   int i,j,min,temp,a[N];
    printf("请输入八个数：\n");
    for (i=0;i<N;i++)
    {   printf("a[%d]=",i);
        scanf("%d",&a[i]);
    }
    printf("\n");
    for (i=0;i<N;i++) printf("%5d",a[i]);
        printf("\n");
    for (i=0;i<N-1;i++)             /* 数据整理 */
    {   min=i;
        for (j=i;j<N;j++)
            if (a[min]>a[j]) min=j;
        temp=a[i];  a[i]=a[min];   a[min]=temp;
    }
    for (i=0;i<N;i++)               /* 输出 */
        printf("%5d",a[i]);
}
```

假设输入的 8 个数据为–5 10 20 40 –100 0 –50 60

该程序的运行结果是_____。

12. 有以下程序：

```
#include <stdio.h>
void main()
{   static char a[5]={'*', '*', '*', '*', '*'};
    int i,j,k;  char space=' ';
    for (i=0;i<5;i++)
    {   printf("\n");
        for (j=1;j<=3*i;j++)  printf("%1c",space);
        for (k=0;k<5;k++)  printf("%3c",a[k]);
    }
}
```

该程序的运行结果是_____。

13. 有以下程序：

```
#include <stdio.h>
void main()
{   char ch[7]={"65ab21"};
    int i,s=0;
    for (i=0;ch[i]>='0'&&ch[i]<= '9';i++)
        s=10*s+ch[i]-'0';
    printf("%d\n",s);
}
```

该程序的运行结果是_____。

14．有以下程序：

```
#include <stdio.h>
#define N  7
void main()
{   int i,j,a[N][N];
    for(i=1;i<N;i++)
    {   a[i][i]=1;
        a[i][1]=1;
    }
    for(i=3;i<N;i++)
    for(j=2;j<=i-1;j++)
        a[i][j]=a[i-1][j-1]+a[i-1][j];
    for(i=1;i<N;i++)
    {   for(j=1;j<=i;j++)
            printf("%6d",a[i][j]);
        printf("\n");
    }
    printf("\n");
}
```

该程序的运行结果是_____。

15．若已定义 int a[10], l;，则以下 fun 函数的功能是：在第一个循环中给前 10 个数组元素依次赋 1, 2, 3, 4, 5, 6, 7, 8, 9, 10；在第二个循环中使 a 数组前 10 个元素中的值对称折叠，变成 1, 2, 3, 4, 5, 5, 4, 3, 2, 1。请填空。

```
fun（int a[ ]）
{   int i,a[11];
    for(i=1;i<=10;i++)  _______=i;
    for(i=0;i<5;i++)  _______=a[i];
    for(i=0;i<10;i++) printf("%3d",a[i]);
}
```

第 6 章　指　　针

一、单项选择题

1．下面是一个初始化指针的语句：int *px=&a;，其中指针变量的名字应该是_____。

A．*pz　　B．a　　C．px　　D．&a

2．若指针 px 为空指针，则_____。

A．px 指向不定　　B．px 的值为零

C．px 的目标为零　　D．px 的地址为零

3．对于语句 int *px[10]，以下说法正确的是_____。

A．px 是一个指针，指向一个数组，数组的元素是整数型

B．px 是一个数组，其数组的每一个元素是指向整数的指针

C．A 和 B 均错，但它是 C 语言的正确语句

D．C 语言不允许这样的语句

4．若有定义语句：int a[2][3], *p[3]; ，则以下语句中正确的是_____。

A．p=a;　　B．p[0]=a;　　C．p[0]=&a[1][2];　　D．p[1]=&a;

5．已知 static int a[]={5,4,3,2,1}，*p[]={a+3,a+2,a+1,a}，**p=q，则表达式* (p[0]+1)+**(q+2)的值是_____。

A．5　　B．4　　C．6　　D．7

6．有以下程序：

```
#include <stdio.h>
void main()
{   int i,j;char a[]="computer",c;
    printf("%s",a);
    for(i=0,j=7;i<j;i++,j--)
    {   c=a[i];
        *(a+i)=*(a+j);
        a[j]=c;
    }
    printf("->%s\n",a);
    c=a[j-1,i=2+j];
    printf("a[%d]=%c\n",i,c);
}
```

程序的运行结果是_____。

A．computer–>computer
a[3] = u

B．computer–>retupmoc
a[5] = m

C．computer–>retupmoc
a[4] = p

D．computer–>retupmoc
a[2] = t

7．有以下程序：

```
#include <stdio.h>
void main()
{   static int a[]={5,6,7,3,2,9};
    int s1,s2,i,*ptr;
    s1=s2=0;
    ptr=&a[0];
    for(i=0;i<5;i+=2)
    {   s1+=*(ptr+i);
        s2+=*(ptr+i+1);
    }
    printf("s1=%d,s2=%d\n",s1,s2);
}
```

程序的运行结果是_____。

A．s1 = 18, s2 = 14　　B．s1 = 14, s2 = 32

C．s1 = 14, s2 = 18　　D．s1 = 15, s2 = 19

8．有以下程序：

```
#include <stdio.h>
void inv(int *x,int n)
{   int *p,t,*i,*j,m=(n-1)/2;
    i=x;
    j=x+n-1;
    p=x+m;
    for(;i<=p;i++,j--)
    {   t=*i;
```

```
        *i=*j;
        *j=t;
    }
}
void main()
{   int i,a[10]={3,7,9,11,0,6,7,5,4,2};
    inv(a,10);
    for(i=0;i<10;i++)
        printf("%d,",a[i]);
}
```

程序的运行结果是_____。

A．0,2,3,4,5,6,7,7,9,11,　　B．11,9,7,7,6,5,4,3,2,0,

C．3,7,9,11,0,6,7,5,4,2,　　D．2,4,5,7,6,0,11,9,7,3,

9．有以下程序：

```
#include <stdio.h>
copy_string(from,to)
char *from,*to;
{   while(*from) *to++=*from++;
    *to='\0';
}
void main()
{   static char s1[ ]="c_program.";
    static char s2[80];
    copy_string(s1,s2);
    printf("%s\n",s2);
    copy_string("123",s2);
    printf("%s\n",s2);
}
```

程序的运行结果是_____。

A．c_program　　B．123　　C．c_program123　　D．c_program 123

123

10．若有以下定义，且 0<= i< 5，则对于 a 数组元素的非法引用是_____。

A．a[i]　　B．*(a+1)　　C．*(p+i)　　D．*(&a+i)

11．有以下函数：

```
int aaa(char *s)
{   char *t=s;
    while(*t++);
      t--;
    return(t-s);
}
```

以下关于 aaa 函数的功能叙述正确的是_____。

A．求字符串 s 的长度　　B．比较两个串的大小

C．将串 s 复制到串 t　　D．求字符串 s 所占字节数

12．下列程序运行后输出的结果为_____。

```
#include <stdio.h>
#include <string.h>
void main()
{   char  *s1="AcDeG";
```

```
    char  *s2="AbdEg";
    s1+=2;   s2+=2;
    printf("%d",strcmp(s1,s2));
}
```

A. 正数　　B. 负数　　C. 零　　D. 不确定的值

13. 下列程序运行后输出的结果为_____。

```
#include <stdio.h>
void main()
{   int  x[5]={2,4,6,8,10},*p,**pp;
    p=x; pp=&p;
    printf("%d", *(p++));
    printf("%3d\n",**pp);
}
```

A. 4　4　　B. 2　4　　C. 2　2　　D. 4　6

14. 下列程序运行的结果为_____。

```
#include <stdio.h>
void main()
{   void abc(char  *p);
    char  str[ ]="cdalb";
    abc(str);
    puts(str);
}
void abc(char  *p)
{   int  i,j;
    for(i=j=0; *(p+i)!='\0';i++)
      if(*(p+i)>='d')
        {   *(p+j)=*(p+i);
            j++;
        }
    *(p+j)='\0';
}
```

A. dl　　B. cd　　C. dalb　　D. c

15. 有以下程序：

```
#include <stdio.h>
void  fun(int  *c,int d)
{   *c=*c+1;d=d+1;
    printf("%c,%c,", *c,d);
}
void main()
{   char  a='A',b='a';
    fun(&b,a);
    printf("%c,%c\n",a,b);
}
```

程序运行后的输出结果是 _____。

A. B,a,B,a　　B. a,B,a,B　　C. A,b,A,b　　D. b,B,A,b

16. 有以下程序：

```
#include <stdio.h>
#include <string.h>
```

```
void main()
{   char *p="abcde\0fghjik\0";
    printf("%d\n",strlen(p));
}
```

程序运行后的输出结果是 _____。

A．12　　B．15　　C．6　　D．5

17．有以下程序：

```
#include <stdio.h>
int fa(int x)
{   return x*x; }
int fb(int x)
{ return x*x*x; }
int f(int (*f1)(),int (*f2)(),int x)
{   return f2(x)-f1(x); }
void main()
{   int i;
    i=f(fa,fb,2);
    printf("%d\n",i);
}
```

程序运行后的输出结果是 _____。

A．–4　　B．1　　C．4　　D．8

18．有以下程序：

```
#include <stdio.h>
void ss(char *s,char t)
{   while(*s)
    {   if(*s==t) *s=t-'a'+'A';
        s++;
    }
}
void main()
{   char str1[100]="abcddfefdbd",c='d';
    ss(str1,c);
    printf("%s\n",str1);
}
```

程序运行后的输出结果是 _____。

A．ABCDDEFEDBD　　B．abcDDfefDbD

C．abcAAfefAbA　　D．Abcddfefdbd

19．有以下程序：

```
#include <stdio.h>
#include <stdlib.h>
void main()
{   char *p,*q;
    p=(char *)malloc(sizeof(char)*20);
    q=p;
    scanf("%s%s",p,q);
    printf("%s  %s\n",p,q);
}
```

若从键盘输入：abc　def<回车>，则输出结果是 _____。

A．def　def　　B．abc　def　　C．abc　d　　D．d　d

20．有以下程序段：

```
int a[10]={1,2,3,4,5,6,7,8,9,10},*p=&a[3],b;
b=p[5];
```

b 中的值是_____。

A．5　　B．6　　C．8　　D．9

21．有以下程序：

```
#include <stdio.h>
void  sum(int *a)
{   a[0]=a[1];}
void main()
{   int aa[10]={1,2,3,4,5,6,7,8,9,10},i;
    for(i=2;i>=0;i--)  sum(&aa[i]);
    printf("%d\n",aa[0]);
}
```

该程序执行后的输出结果是_____。

A．4　　B．3　　C．2　　D．1

22．有以下定义：

```
#include <stdio.h>
char a[10],*b=a;
```

不能给数组 a 输入字符串的语句是_____。

A．gets(a);　　B．gets(a[0]);　　C．gets(&a[0]);　　D．gets(b);

23．有以下程序：

```
#include <stdio.h>
int a=2;
int f(int *a)
{   return (*a)++;  }
void main()
{   int s=0;
    {   int a=5;
        s+=f(&a);
    }
    s+=f(&a);
    printf("%d\n",s);
}
```

程序执行后的输出结果是_____。

A．10　　B．9　　C．7　　D．8

24．有以下程序：

```
#include <stdio.h>
void main()
{   int a[3][3],*p,i;
    p=&a[0][0];
    for(i=0;i<9;i++)
        p[i]=i;
```

```
    for(i=0;i<3;i++)
        printf("%d",a[1][i]);
}
```

该程序执行后的输出结果是_____。

A．012　　B．123　　C．234　　D．345

25．有以下程序：

```
#include <stdio.h>
void main()
{   int a[3][2]={0},(*ptr)[2],i,j;
    for(i=0;i<2;i++)
        {   ptr=a+i;
            scanf("%d",ptr);
            ptr++;
        }
    for(i=0;i<3;i++)
      { for(j=0;j<2;j++)
          printf("%2d",a[i][j]);
        printf("\n");
      }
}
```

若运行时输入：1 2 3<回车>，则输出结果是_____。

A．产生错误信息

B．1 0
2 0
0 0

C．1 2
3 0
0 0

D．1 0
2 0
3 0

26．以下语句或语句组中，能正确进行字符串赋值的是_____。

A．char *sp; *sp="right!";　　B．char s[10];s="right!";

C．char s[10]; *s="right!";　　D．char *sp="right!";

27．有以下程序：

```
#include <stdio.h>
void main()
{   char s[]="159",*p;
    p=s;
    printf("%c",*p++);printf("%c",*p++);
}
```

该程序执行后的输出结果是_____。

A．15　　B．16　　C．12　　D．59

28．若在定义语句：int ,b,c, *p=&c;之后，接着执行以下选项中的语句，则能正确执行的语句是 _____。

A．`scanf("%d",a,b,c);`　　B．`scanf("%d%d%d",a,b,c);`

C．`scanf("%d",p);`　　D．`scanf("%d",&p);`

29．程序中若有如下说明和定义语句：

```
#include <stdio.h>
char fun(char *);
void main()
```

```
{   char *s="one",a[5]={0},(*f1)()=fun,ch;
    ......
}
```

以下选项中对函数 fun 的正确调用语句是_____。

A. (*f1)(a);　　B. *f1(*s);　　C. fun(&a);　　D. ch = *f1(s);

30. 有以下程序：

```
#include <stdio.h>
#include  <string.h>
void main(int  argc,char  *argv[])
{   int  i,len=0;
    for(i=1;i<argc;i++)
      len+=strlen(argv[i]);
    printf("%d\n",len);
}
```

程序编译连接后生成的可执行文件是 ex1.exe，若运行时输入带参数的命令行是：

```
ex1   abcd    efg     10<回车>
```

则运行的结果是：_____。

A. 22　　B. 17　　C. 12　　D. 9

二、填空题

1. 以下程序所列函数 replace(char *s1,char *s2,char str1,char *str2)的功能是将已知字符串 s1 中的所有与字符串 str1 相同的子串替换成字符串 str2，并将替换后所生成的新的字符串存于字符数组 s2 中。

说明：生成字符串 s2 的过程是一个循环，顺序访问字符串 s1 的每个字符；当从某个字符开始不能构成与 str1 相同的子字符时，就把该字符复制到数组 s2，当从某个字符开始能构成一个与 str1 相同的子字符串时，就将字符串 str2 的各字符复制到字符数组 s2，并继续访问字符串 s1 中那个子串之后的字符，直至字符串 s1 被访问完毕，字符复制即告结束。

下列程序运行的结果是输出：

```
CBCXYZdefg  abABCXYZd  abab
```

程序如下：

```
#include <stdio.h>
void replace(char *s1,char *s2,char *str1,char *str2)
{   char  *t0,*t1,*t2;
    while(_____)
        { for(t0=s1,t1=str1;*t1!='\0'&&_____;t0++,t1++);
                if(*t1!='\0')
                   *s2++=_____;
                else
                    {  for(t1=str2;*t1!='\0';)
                        *s2++=_____;
                        _____;
                    }
        }
    *s2='\0';
}
void main()
```

```
{   char s1[]="abcdefg  ababcd  abab.";
    char s2[80];
    replace(s1,s2,"abc","ABCXYZ");
    printf("%s\n",s2);
}
```

2．下面的程序可将已安排好序的两个字符串 a 和 b 中的字符降序归并到字符串 c 中。请填空。

```
#include <stdio.h>
#include <string.h>
void main()
{   char  a[ ]="mkigeca";
    char  b[ ]="zyponljhdb";
    char  c[80],*p;
    int  i=0,j=0,k=0;
    while(a[i]!='\0'&&b[j]!='\0')
        {   if(a[i]<b[j])
                { _______; j++; }
            else
                { ______; i++; }
            ++k;
        }
    c[k]='\0';
    if( _______ )
        p=b+j;
    else
        _______ ;
    ________;
    puts(c);
}
```

3．设有定义：int　n,*k = &n;，以下语句将利用指针变量 k 读写变量 k 和 n 中的内容，请将语句补充完整。

```
scanf("%d", ___________);
printf("%d\n", _________);
```

4．fun 函数的功能是：首先对 a 所指的 N 行 N 列的矩阵，找出各行中的最大的数，再求这 N 个最大值中的最小的那个数作为函数值返回。请填空。

```
#include <stdio.h>
#define N 100
int  fun(int(*a)[N])
{   int  row,col=0,max,min;
    for(row=0;row<N;row++)
    {   for(max=a[row][0],col=1;col<N;col++)
          if(_______)max=a[row][col];
        if(row==0)min=max;
        else if(______) min=max;
    }
    return  min;
}
```

5. 函数 sstrcmp( )功能是对两个字符串进行比较。当 s 所指字符串和 t 所指字符串相等时，

返回值为 0；当 s 所指字符串大于 t 所指字符串时，返回值大于 0；当 s 所指字符串小于 t 所指字符串时，返回值小于 0（功能等同于库函数 strcmp()）。请填空。

```
#include <stdio.h>
int sstrcmp(char *s,char *t)
{   while(*s && *t && *s==_____ )
      { s++;t++;  }
    return___;
}
```

6．下列程序的运行结果是_____。

```
#include <stdio.h>
void swap(int *a,int *b)
{   int *t;
    t=a; a=b; b=t;
}
void main()
{   int x=3,y=5,*p=&x,*q=&y;
    swap(p,q);
    printf("%d %d\n",*p,*q);
}
```

7．以下 sstrcpy()函数实现字符串复制，即将 t 所指字符串复制到 s 所指向内存空间中，形成一个新的字符串 s。请填空。

```
#include <stdio.h>
void sstrcpy(char *s,char *t)
{   while(*s++=_____); }
void main()
{   char str1[100],str2[]="abcdefgh";
    sstrcpy(str1,str2);
    printf("%s\n",str1);
}
```

8．下列程序的运行结果是_____。

```
#include <string.h>
char *ss(char *s)
{   return s+strlen(s)/2; }
void main()
{   char *p,*str="abcdefgh";
    p=ss(str);
    printf("%s\n",p);
}
```

9．以下程序中，函数 huiwen( )的功能是检查一个字符串是否是回文，当字符串是回文时，函数返回字符串：yes!，否则函数返回字符串：no!，并在主函数中输出，所谓回文即正向与反向的拼写都一样，如 adgda。请填空。

```
#include <stdio.h>
#include <string.h>
char *huiwen(char *str)
{   char *p1,*p2;
    int i,t=0;
    p1=str;p2=_____;
```

```
    for(i=0;i<=strlen(str)/2;i++)
      if(*p1++!=*p2--){  t=1;break;  }
      if(_____________) return("yes!");
      else return("no!");
}
void main()
{   char str[50];
    printf("Input:"); scanf("%s",str);
    printf("%s\n", _____________);
}
```

10. 以下程序的输出结果是_____。

```
#include <stdio.h>
#include <string.h>
char *fun(char *t)
{   char *p=t;
    return(p+strlen(t)/2);
}
void main()
{   char *str="abcdefgh";
    str=fun(str);
    puts(str);
}
```

11. 以下程序中，给指针 p 分配三个 double 型动态内存单元，请填空。

```
#include <stdio.h>
#include <stdlib.h>
void main ()
{   double *p;
    p=(double *)malloc(_____________);
    p[0]=1.5;p[1]=2.5;p[2]=3.5;
    printf("%f %f %f\n",p[0],p[1],p[2]);
}
```

12. 有以下程序：

```
#include <stdio.h>
void fun(int *a,int n) /* fun 函数的功能是将 a 所指数组元素从大到小排序 */
{   int t,i,j;
    for(i=0;i<n-1;i++)
        for(j=i+1;j<n;j++)
            if (a[i]<a[j])
                {t=a[i];a[i]=a[j];a[j]=t;}
}
void main()
{   int c[10]={1,2,3,4,5,6,7,8,9,0},i;
    fun(c+4,6);
    for (i=0;i<10;i++)
        printf("%d,",c[i]);
    printf("\n");
}
```

该程序运行后的输出结果是 _____。

13. 有以下程序：

```
#include <stdio.h>
void fun(char **p)
{   ++p; printf("%s\n",*p); }
void main()
{   char *a[]={"Morning","Afternoon","Evening","Night"};
    fun(a);
}
```

该程序运行后的输出结果是_____。

14. 有以下程序：

```
#include <stdio.h>
void point(char *p)
{   p+=3;  }
void main()
{   char b[4]={'a','b','c','d'},*p=b;
    point(p);printf("%c\n",*p);
}
```

该程序运行后的输出结果是_____。

15. 有以下程序：

```
#include <stdio.h>
void f(int *x,int *y)
{   int t;
    t=*x; *x=*y; *y=t;
}
void main()
{   int a[8]={1,2,3,4,5,6,7,8},i, *p, *q;
    p=a;q=&a[7];
    while(p<q)
        {   f(p,q);p++;q--;  }
    for(i=0;i<8;i++)
        printf("%d,",a[i]);
}
```

该程序运行后的输出结果是_____。

第 7 章　模块化程序设计

一、单项选择题

1. C 源程序的基本单位是_____。

A. 程序行　　B. 语句　　C. 函数　　D. 字符

2. 下列说法不正确的是_____。

A. C 程序由若干个源文件组成，一个源文件由若干个函数组成

B. #include 和#define 不是 C 语句

C. APH 和 aph 是两个不同的变量

D. 当输入数据时，对于整型变量只能输入整型值；对于实型变量只能输入实型值

3. 有以下程序：

```
#include <stdio.h>
void fun2(char a,char b)
```

```
{   printf("%c%c ", a,b);  }
char a='A',b='B';
void fun1()
{   a='C';b='D';  }
void main()
{   fun1();
    printf("%c%c ", a,b);
    fun2('E', 'F');
}
```

程序运行后的输出结果是_____。

A．CD EF　　B．AB EF　　C．AB CD　　D．CD AB

4．以下叙述中正确的是_____。

A．全局变量的作用域一定比局部变量的作用域范围大

B．静态（static）类型变量的生存期贯穿于整个程序的运行期间

C．函数的形参都属于全局变量

D．未在定义语句中赋初值的 auto 变量和 static 变量的初值都是随机值

5．在一个 C 源程序文件中所定义的全局变量，其作用域为_____。

A．所在文件的全部范围

B．所在程序的全部范围

C．所在函数的全部范围

D．由具体定义位置和 extern 说明来决定范围

6．C 语言中，局部变量缺省的存储类型是_____。

A．auto　　B．static　　C．register　　D．extern

7．有以下程序：

```
#include <stdio.h>
fun(int x,int y)
{   static int m=0,i=2;
    i+=m+1;
    m=i+x+y;
    return  m;
}
void main()
{   int j=1，m=1，k;
    k=fun(j,m);
    printf("%d,", k);
    k=fun(j,m);
    printf("%d\n",k);
}
```

程序运行后的输出结果是_____。

A．5,5　　B．5,11　　C．11,11　　D．11,5

8．有以下程序：

```
#include <stdio.h>
int a=2;
int f(int n)
{   static int a=3;
    int t=0;
    if(n%2)
```

```
    {   static int a=4;t+=a++;  }
    else
    {   static int a=5; t+=a++;  }
    return  t+a++;
}
void main()
{   int s=a, i;
    for(i=0;i<3; i++) s+=f(i);
    printf("%d\n", s);
}
```

程序运行后的输出结果是_____。

A．26　　B．28　　C．29　　D．24

9．有以下程序：

```
#include <stdio.h>
int c=1;
void func();
void main()
{   static int a=5;int b=6;
    printf("a=%d,b=%d,c=%d\n",a,b,c);
    func();
    printf("a=%d,b=%d,c=%d\n",a,b,c);
    func();
}
void func()
{   static int a=4;
    int b=10;
    a+=2;
    c+=10;
    b+=c;
    printf("a=%d,b=%d,c=%d\n",a,b,c);
}
```

程序运行后的输出结果是_____。

A．a＝5,b＝6, c＝1
　　a＝6,b＝21,c＝11
　　a＝5,b＝6, c＝11
　　a＝8,b＝31, c＝21

B．a＝5,b＝6, c＝1
　　a＝7,b＝17,c＝11
　　a＝5,b＝6, c＝11
　　a＝9,b＝17, c＝11

C．a＝5,b＝6, c＝1
　　a＝6,b＝21,c＝11
　　a＝6,b＝21, c＝11
　　a＝8,b＝31, c＝21

D．a＝5,b＝6, c＝1
　　a＝7,b＝17,c＝11
　　a＝7,b＝17, c＝11
　　a＝9,b＝38, c＝21

10．以下只有在使用时才为该类型变量分配内存的存储类型是_____。

A．auto 和 static　　B．register 和 static

C．auto 和 register　　D．extern 和 register

11．如果在一个函数中的复合语句中定义了一个变量，则该变量是_____。

A．只在该复合语句中有效　　B．在该函数中有效

C．在本程序范围内均有效　　D．为非法变量

12. 以下程序的输出结果是_____。

```
#include <stdio.h>
#define f(x,y) (x)* (y)
void main()
{   int a=3, b=4;
    printf("%d\n", f(a++,b++));
}
```

A. 12　　B. 15　　C. 16　　D. 20

13. 以下程序的输出结果是_____。

```
#include <stdio.h>
int  func(int  a,int b);
void main()
{   int k=4,m=1,p;
    p=func(k,m);
    printf("%d,",p);
    p=func(k,m);
    printf("%d\n",p);
}
int func(int  a,int b)
{   static int  m=0, i=2;
    i +=m+1;
    m=i+a+b;
    return(m);
}
```

A. 8,17　　B. 8,16　　C. 8,20　　D. 8,8

14. 以下叙述中正确的是_____。

A. 预处理命令行必须位于源文件的开头

B. 在源文件的一行上可以有多条预处理命令

C. 宏名必须用大写字母表示

D. 宏替换不占用程序的运行时间

15. 有以下程序：

```
#include <stdio.h>
#define  P  3
void f(int x)
{   return(P*x*x);  }
void main()
{   printf("%d\n",f(3+5));  }
```

程序运行后的输出结果是_____。

A. 192　　B. 29　　C. 25　　D. 编译出错

16. 有以下程序：

```
#include <stdio.h>
#define  N  5
#define  M  N+1
#define  f(x)  (x*M)
void main()
{   int i1,i2;
    i1= f(2);
```

```
    i2= f(1+1);
    printf("%d  %d\n",i1,i2);
}
```

程序运行后的输出结果是_____。

A．12　12　　B．11　7　　C．11　11　　D．12　7

17．有以下程序：

```
#include <stdio.h>
#define f(x) (x*x)
void main()
{   int i1, i2;
    i1=f(8)/f(4);
    i2=f(4+4)/f(2+2);
    printf("%d, %d\n",i1,i2);
}
```

程序运行后的输出结果是_____。

A．68, 28　　B．4, 4　　C．4, 3　　D．64, 64

18．在宏定义# define E 2.71828 中，宏名 E 代替一个_____。

A．单精度数　　B．双精度数

C．常数　　D．一个字符串

19．在宏定义# define E 2.71828 中，在其后程序中使用 2.71828，可以用_____表示。

A．E　　B．e　　C．EXP　　D．EXP( )

20．有以下程序：

```
# define  PI 3.141593
# include <stdio.h>
void main()
{   printf ("PI=%f\n",PI);  }
```

程序运行结果为_____。

A．3.141593 = 3.141593　　B．PI = 3.141593

C．3.141593 = PI　　D．程序有错误，无结果

21．有宏定义：

```
#define   MULT1(a,b)    a*b
#define   MULT2 (a,b)    (a)*(b)
```

在后面程序中有宏引用：y = MULT1(3+2, 5+8);　z = MULT2(3+2, 5+8);

则 y 和 z 的值是_____。

A．y = 65, z = 65　　B．y = 21, z = 65

C．y = 65, z = 21　　D．y = 21, z = 21

22．以下程序中的输出结果是_____。

A．5　　B．6　　C．9　　D．8

```
#include <stdio.h>
#define  N  2
#define  M  N+1
#define  NUM  ( M+1 )*M/2
void main()
{   int i, n=0;
    for(i=1;i<=NUM;i++)
```

```
        n++;
    printf("%d\n",i);
}
```

23. 格式 1　#include　"文件名"

格式 2　#include　<文件名>

下面 4 个结论，_____是正确的。

A．格式 1 中的文件名能带路径，而格式 2 不能

B．格式 1 中的文件名不能带路径，而格式 2 能

C．如果被包含文件在当前目录下，两种格式都能用

D．如果文件名中不带路径，格式 1 能搜索当前目录和 C 编译环境指定的标准方式检查其他目录，而格式 2 不行

24. 有以下程序：

```
#include <stdio.h>
#define  MIN(x,y)  (x)<(y)?(x):(y)
void main()
{   int i,j,k;
    i=10;
    j=15;
    k=10*MIN(i,j);
    printf("%d\n",k);
}
```

程序运行的输出结果是_____。

A．15　　B．100　　C．10　　D．150

25. C 编译系统对宏命令的处理是_____。

A．和其他 C 语句同时进行编译的　　B．在对其他成分正式编译之前处理的

C．在程序执行时进行代换处理的　　D．在程序连接时处理的

26. 有#define N 80 和 printf("*******"); 以下说法中正确的是_____。

A．两者都是 C 语句　　B．前者是 C 语句，后者不是

C．前者不是，而后者是 C 语句　　D．两者都不是 C 语句

27. 有以下程序：

```
#include <stdio.h>
#define f(x)  x*x
void main()
{   int a=6,b=2,c;
    c=f(a)/f(b);
    printf("%d\n",c);
}
```

程序运行的的输出结果是_____。

A．9　　B．6　　C．36　　D．18

28. 以下程序的输出结果是_____。

```
#include <stdio.h>
#define  FUDGF(y)  2.84+y
#define  PR(a)  printf("%d",(int )a)
#define  PRINT1(a)  PR(a);  putchar('\n')
```

```
void main()
{   int  x=2;
    PRINT1 ( FUDGF(5)*x);
}
```

A．11　　B．12　　C．13　　D．15

29．头文件的扩展名可以是_____。

A．.h 和.obj　　B．.h 和.c　　C．.c 和.obj　　D．任意

30．以下叙述正确的是_____。

A．若一些源程序中包含某些头文件，当该头文件有错时，只需对该头文件进行修改，包括此头文件所有源程序不必重新进行编译

B．用#include 包含的头文件的后缀不可以是".a"

C．宏命令行可以看做是一行 C 语句

D．C 编译中的预处理是在编译之前进行的

二、填空题

1．宏展开是在_____时进行的，不分配_____，不进行值的_____处理。

2．宏替换不占_____时间，只占_____时间。

3．以下程序的输出结果是 _____。

```
#define  PR(ar)  printf("ar=%d  ",ar)
#include <stdio.h>
void main()
{   int j,a[]={1,3,5,7,9,11,13,15},*p=a+5;
    for(j=3;j;j++)
        switch( j )
        {   case 1:
            case 2 : PR(*p++);break;
            case 3 :PR(*(--p));
        }
}
```

4．文件包含使用_____命令，其作用是将一已知文件的内容_____到命令处。

5．写出判断 x 是否是奇数的宏定义，宏名为 isodd: _____。

6．C 语言的编译预处理命令主要有三类：_____、_____、_____。

7．若所需头文件 file1.h 恰巧在源文件目录下，则应用_____形式来实现。

8．函数的形式参数的作用域是_____；全局变量与函数体内定义的局部变量重名时，_____变量优先。

9．执行以下程序后的输出结果是_____。

```
#include <stdio.h>
int  k=0;
void fun(int  m)
{   m+=k; k+=m;  printf("m=%d  k=%d\n",m,k++);  }
void main()
{   int  i=4;
    fun(i++);
    printf("i=%d  k=%d\n",i,k);
}
```

10．以下程序的输出结果是_____。

```
#include <stdio.h>
#define  FUN(y)  3.14159+y
#define  PR(a)  printf("%d",(int)(a))
#define  PRINT1(a)  PR(a);putchar('\n')
void  main()
{  int x=3;
   PRINT1(FUN(4)*x);
}
```

11．下面程序由两个源程序文件：t4.h 和 t4.c 组成，程序编译运行的结果是_____。

t4.h 的源程序为

```
#define  N  10
#define  f2(x)  (x*N)
```

t4.c 的源程序为

```
#include <stdio.h>
#define  M  8
#define  f(x)  ((x)*M)
#include "t4.h"
void main()
{   int  i,j;
    i=f(1+1);
    j=f2(1+1);
    printf("%d  %d\n",i,j);
}
```

12．执行以下程序后的输出结果是_____。

```
#include <stdio.h>
void fun()
{   static int a=0;
    a+=2;
    printf("%d\n",a);
}
void main()
{   int cc;
    for(cc=1;cc<4;cc++)  fun();
    printf("\n");
}
```

13．有如下程序：

```
#define POWER(x)   x*x
#include <stdio.h>
void main()
{   int    x=5, y=3;  int    t, z;
    t=POWER(x+y);
    z=POWER(x*y);
    printf("%d,  %d\n",t,z);
}
```

该程序的运行结果为_____。

14．有如下程序：

```
#include <stdio.h>
```

```
void main()
{   int b=5;
    #define  b   2
    #define  f(x)  b*(x)
    int  y=3;   printf("%d\n",f(y+1));
    #undef  b
    printf("%d\n",f(y+1));
    #define  b  3
    printf("%d\n",f(y+1));
}
```

该程序的运行结果为_____。

15．求数组中的最大元素，调试程序如下，请填空。

```
#include <stdio.h>
_________N  10
#define  TEST  1
void  main()
{   int i,max,a[N];
    # if  TEST
    for (i=0; i<_______,i++)  a[i]=10+i;
    # else
    for (i=0; i<N; i++) scanf("%d",&a[i]);
    ________
    max=_______ ;
    for (i=1;i<N;i++)
    if (max______a[i])  max=a[i];
    printf("Max=%d\n",max);
}
```

第 8 章　结　构　体

一、单项选择题

1．联合体成员的数据类型_____。

A．相同　　B．可以不同也可以相同　　C．长度一样　　D．是结构体变量

2．已知

```
struct student
{   char *name;
    int student_n;
    char grade;
};
struct student temp, *p=&temp;
temp.name="chou";
```

则下面不正确的是_____。

表达式	值
A．p –>name	chou
B．(*p) –>name+2	h
C．*p –>name+2	e
D．* (p –>name+2)	o

3．设有以下语句：

```
struct  st{int  n; struct  st  *next;};
static  struct  st  a[3]={5,&a[1],7,&a[2],9,'\0'},*p;
p=&a[0];
```

则表达式_____的值是 6。

A．p++ –>n　　B．p–>n++　　C．(*p).n++　　D．++p –>n

4．当说明一个结构体变量时，系统分配给它的内存是_____。

A．各成员所需的内存量的总和

B．结构中第一个成员所需的容量

C．成员中最后一个成员所需内存量

D．成员中所占内存量最大者所需的容量

5．设有如下定义：

```
struct  sk
{   int   n;
    float  x;
}date，*p;
```

若要使 p 指向 date 中的 n 域，正确的赋值语句是_____。

A．p = &date.n;　　B．*p = date.n;

C．p = (struct　sk *)&date.n　　D．p = (struct　sk *)date.n;

6．下列程序运行结果为_____。

```
#include <stdio.h>
void main()
{   union
    {   int  a;
        long b;
        unsigned   char  c;
    }m;
    m.b=0x12345678;
    printf("%x\n",m.a);
    printf("%x\n",m.c);
}
```

A．1234　　B．5678　　C．12345678　　D．0

　　12　　78　　1234　　5678

7．有以下程序：

```
#include <stdio.h>
struct   STU
{   char   num[10];   float   score[3]; };
#include <stdio.h>
void main()
{   struct STU  s[3]={{"20021",90,95,85},{"20022",95,80,75},
                      {"20023",100,95,90}},*p=s;
    int i; float  sum=0;
    for(i=0;i<3;i++)
      sum=sum+p->score[i];
    printf("%6.2f\n",sum);
}
```

程序运行后的输出结果是_____。

A．260.00　　B．270.00　　C．280.00　　D．285.00

8．设有如下定义：

```
struct  sk
{   int a;
    float b;
}data;
int *p;
```

若要使 p 指向 data 中的 a 域，正确的赋值语句是_____。

A．p＝&a;　　B．p＝data.a　　C．p＝&data.a;　　D．*p = data.a;

9．若有以下说明和定义：

```
typedef  int  *INTEGER;
INTEGER  p, *q;
```

以下叙述正确的是_____。

A．p 是 int 型变量

B．p 是基类型为 int 的指针变量

C．q 是基类型为 int 的指针变量

D．程序中可用 INTEGER 代替 int 类型名

10．若程序中有下面的说明和定义：

```
struct  abc{int  x; char  y;}
struct  abc  s1,s2;
```

则会发生的情况是_____。

A．编译出错　　B．程序将顺利编译、连接、执行

C．能顺利通过编译、连接，但不能执行　　D．能顺利通过编译，但连接出错

11．设有以下语句：

```
typedef struct  S
{  int g;  char  h;}T;
```

则下面叙述中正确的是_____。

A．可用 S 定义结构体变量　　B．可以用 T 定义结构体变量

C．S 是 struct 类型的变量　　D．T 是 struct S 类型的变量

12．有以下程序：

```
#include <stdio.h>
struct STU
{   char name[10];
    int num;
    int Score;
};
void main()
{   struct STU   s[5]={{"YangSan",20041,703},{"LiSiGuo",20042,580},
                       {"wangYin",20043,680},{"SunDan",20044,550},
                       {"Penghua",20045,537}},*p[5], *t;
    int i,j;
    for(i=0;i<5;i++)   p[i]=&s[i];
```

```
    for(i=0;i<4;i++)
    for(j=i+1;j<5;j++)
        if(p[i]->Score>p[j]->Score)
            {   t=p[i];p[i]=p[j];p[j]=t;  }
        printf("%d  %d\n",s[1].Score,p[1]->Score);
}
```

程序执行后输出结果是_____。

A．550　550　　B．680　680　　C．580　550　　D．580　680

13．若有以下说明和定义：

```
union dt
{   int a;char b;double c;  }data;
```

以下叙述中错误的是_____。

A．data 的每个成员起始地址都相同

B．变量 data 所占的内存字节数与成员 c 所占字节数相等

C．程序段：data.a=5;printf("%f\n",data.c);输出结果为 5.000000

D．data 可以作为函数的实参

14．设有如下说明：

```
typedef struct ST
{   long a;int b;char c[2];}NEW;
```

则下面叙述中正确的是_____。

A．以上的说明形式非法　　B．ST 是一个结构体类型

C．NEW 是一个结构体类型　　D．NEW 是一个结构体变量

15．以下对结构体类型变量 td 的定义中，错误的是_____。

A．
```
typedef struct aa
    {   int n;
        float m;
    }AA;
  AA td;
```

B．
```
struct aa
    {   int n;
        float m;
    };
 struct aa td;
```

C．
```
struct
    {   int n;
        float m;
    }aa;
    struct aa td;
```

D．
```
struct
    {   int n;
        float m;
    }td;
```

16．设有一结构体类型变量，定义如下：

```
struct date
{   int year;
    int month;
    int day;
};
struct worklist
{   char name [20];
    char sex;
    struct date birthday;
```

```
}person;
```

若要对结构体变量 person 的出生年份进行赋值时，下面正确的赋值语句是 _____。

A．year=1976;　　B．birthdat.year=1976;

C．person.birthday.year=1976;　　D．person.year=1976;

17．以下关于 typedef 的叙述错误的是_____。

A．用 typedef 可以增加新类型

B．typedef 只是将已存在的类型用一个新的名字来代表

C．用 typedef 可以为各种类型说明一个新名，但不能用来为变量说明一个新名

D．用 typedef 为类型说明一个新名，通常可以增加程序的可读性

18．有以下程序 ：

```
#include <stdio.h>
struct tt
{  int x;
   struct tt *y; } *p;
 struct tt a[4]={20,a+1,15,a+2,30,a+3,17,a};
void main()
{  int i;
   p=a;
   for(i=1;i<=2;i++)
   {  printf("%d,",p->x); p=p->y;   }
}
```

程序的运行结果是_____。

A．20, 30,　　B．30, 17,　　C．15, 30,　　D．20, 15,

19．有以下程序：

```
#include <stdio.h>
#include <string.h>
typedef struct{ char name[9];char sex; float score[2]; } STU;
STU f(STU a)
{   STU b={"Zhao",'m',85.0,90.0}; int i;
    strcpy(a.name,b.name);
    a.sex=b.sex;
    for(i=0;i<2;i++)
      a.score[i]=b.score[i];
    return a;
}
void main()
{   STU c={"Qian",'f',95.0,92.0},d;
    d=f(c);
    printf("%s,%c,%2.0f,%2.0f\n",d.name,d.sex,d.score[0],d.score[1]);
}
```

程序的运行结果是_____。

A．Qian, f, 95, 92　　B．Qian, m, 85, 90

C．Zhao, m, 85, 90　　D．Zhao, f, 95, 92

20．以下程序的输出结果是_____。

```
#include <stdio.h>
union myun
```

```
{   struct  { int x，y,z; }u;
    int k;}a;
void main（）
{   a.u.x=4;a.u.y=5;a.u.z=6;
    a.k=0;
    printf（"%d\n"，a.u.x）;
}
```

A．4　　B．5　　C．6　　D．0

二、填空题

1．已知学生的记录由学号和学习成绩构成，N 名学生的数据已存入结构体数组 a 中。请编写函数 fun，函数的功能是找出成绩最高的学生记录，通过形参返回主函数（规定只有一个最高分）。已给出主函数和函数 fun 的首部，请在函数 fun 的花括号中填入你编写的若干语句。

```
#include <stdio.h>
#include <string.h>
#include <conio.h>
#define N 10
typedef struct ss
{   char num[10]; int s; } STU;
fun( STU a[], STU *s)
{ _____ }
void main()
{   STU a[N]={ {"A01",81},{"A02",89},{"A03",66},{"A04",87},{"A05",77},
          {"A06",90},{"A07",79},{"A08",61},{"A09",80},{"A10",71} }, m;
    int i;
    printf("**** the original data ****\n");
    for(i=0;i<N; i++)  printf("N0=%s Mark=%d\n", a[i].num,a[i].s);
    fun( a,&m);
    printf("**** the result****\n");
    printf(" the top: %s ,%d\n", m.num,m.s);
}
```

2．学生的记录由学号和成绩组成，N 名学生的数据已在主函数中放入结构体数组 s 中。请编写函数 fun，它的功能是把成绩最低的学生数据放在 h 所指的数组中。注意：最低分可能不止一个学生，函数返回成绩最低的学生的个数。请在函数 fun 的花括号中填入你编写的若干语句。

```
#include <stdio.h>
#define N 16
typedef struct
{   char num[10];
    int s;
}STREC;
int fun ( STREC *a, STREC *p )
{ _____ }
void main()
{   STREC s={{"GA05",85},{"GA03",76},{" GA02",69},{" GA04",85},
             {" GA01",91},{" GA07",72},{" GA08",64},{" GA06",87},
             {" GA015",87},{" GA013",64},{" GA012",87},{" GA014",87},
```

```
                {" GA011",87},{" GA017",87},{" GA018",64},{" GA016",87}};
    STREC h[N];
    int i,n;
    n=fun(s,h);
    printf("The %d lowest score :\n",n");
    for(i=0;i<n;i++)
        printf("%s  %4d\n",h[i].num,h[i].s);
    printf("\n");
}
```

3. 某学生的记录由学号、8 门课程成绩和平均分组成，学号和 8 门课程的成绩已在主函数中给出。请编写函数 fun，它的功能是求出该学生的平均分并放在记录的 ave 成员中。请自己定义正确的形参。

例如，假定学生的成绩为 85.5, 76, 69.5, 85, 91, 72, 64.5, 87.5，则他的平均分应当为 78.875。

```
#include <stdio.h>
#define N 8
typedef struct
{   char num[10];
    double s[N];
    double ave;
}STREC;
void fun(_______)
{  _________   }
void main()
{   STREC s={"GA005",85.5,76,69.5,85,91,72,64.5,87.5};
    int i;
    fun(&s);
    printf("The %s\'s student data:\n ",s.num);
    for(i=0;i<8;i++)
      printf("%4.1f\n",s.s[i]);
    printf("\nave=%7.3f\n",s.ave);
}
```

4. 程序如下：

```
#include <stdio.h>
struct n_c{  int  x; char  c;  };
void func(struct  n_c  b)
{   b.x=20;  b.c='y';  }
void main()
{   struct  n_c  a={10,'x'};
    func(a);
    printf("%d%c",a.x,a.c);
}
```

该程序的运行结果是_____。

5. 下面程序是从键盘上输入一个日期，计算该日是该年中第几天，并从屏幕上显示出来。

```
#include <stdio.h>
struct  date{int  d; int  m; int  y;};
int  days(struct  date  pd);
void  main()
```

```
{   struct  date  ymd;
    printf("Enter year-month-day:");
    scanf("%d-%d-%d",&ymd.y,&ymd.m,&ymd.d);
    printf("The Passed days:%d\n",days(_______));
    }
int  days(struct  date  pd)
{   static int tab[2][13]={{0,31,28,31,30,31,30,31,31,30,31,30,31},
                           {0,31,29,31,30,31,30,31,31,30,31,30,31}};
    int  number,i,lp;
    number=pd.d;
    lp=pd.y%4==0 && pd.y%100!=0||pd.y%400==0;
    for (i=1; i<pd.m;i++)  number= ___________;
    return(_________);
}
```

第 9 章　位　运　算

一、单项选择题

1．设有定义的语句：char c1=92,c2=92;，则以下表达式中值为零的是_____。

A．c1^c2　　B．c1&c2　　C．~c2　　D．c1|c2

2．设 char 型变量 x 中的值为 10100111，则表达式(2＋x)^(~3)的值是_____。

A．10101001　　B．10101000　　C．11111101　　D．01010101

3．整型变量 x 和 y 的值相等，且为非 0 值，则以下选项中，结果为零的表达式是_____。

A．x||y　　B．x|y　　C．x&y　　D．x^y

4．设 int b＝2;表达式(b>>2)/(b>>1)的值是_____。

A．0　　B．2　　C．4　　D．8

5．printf("%d\n", 12&012);的输出结果是_____。

A．12　　B．8　　C．6　　D．012

6．设有如下定义：int x＝1，y＝–1;，则 printf("%d\n"，(x– –&++y));的输出结果是_____。

A．1　　B．0　　C．–1　　D．2

7．有以下程序：

```
#include <stdio.h>
void main()
{   int  x=3, y=2,z=1;
    printf("%d\n",x/y&~z);
}
```

程序运行后的输出结果是_____。

A．3　　B．2　　C．1　　D．0

8．以下程序的功能是进行位运算。

```
#include <stdio.h>
void main()
{   unsigned char  a, b;
```

```
    a=7^3; b=~4&3;
    printf("%d %d\n",a,b);
}
```

程序运行后的输出结果是_____。

A．4 3　　B．7 3　　C．7 0　　D．4 0

9．以下程序段的运行结果是_____。

```
unsigned a=0356,b;
b=~a|a<<2+1;
printf("%x\n",b);
```

A．ffba　　B．ff71　　C．fff8　　D．fc02

10．若变量已正确定义，则以下语句的输出结果是_____。

```
s=32; s^=32; printf("%d",s);
```

A．–1　　B．0　　C．1　　D．32

11．以下程序运行后的输出结果是_____。

```
#include <stdio.h>
void main()
{   int c=35;
    printf("%d\n",c&c);
}
```

A．0　　B．70　　C．35　　D．1

12．设有以下语句，执行后 c 的值是_____。

```
int a=1,b=2,c;
c=a^(b<<2);
```

A．6　　B．7　　C．8　　D．9

13．以下程序的输出结果是_____。

```
#include <stdio.h>
void main()
{   int x=0.5;
    char z='a';
    printf("%d\n", (x&1)&&(z<'z') );
}
```

A．0　　B．1　　C．2　　D．3

14．以下程序的输出结果是_____。

```
#include <stdio.h>
void main()
{   char x=040;
    printf("%o\n",x<<1);
}
```

A．100　　B．80　　C．64　　D．32

15．有以下程序：

```
#include <stdio.h>
void main()
```

```
{   unsigned char a,b,c;
    a=0x3; b=a|0x8; c=b<<1;
    printf("%d  %d\n",b,c);
}
```

程序运行后的输出结果是_____。

A．–11　12　　B．–6　–13　　C．12　24　　D．11　22

二、填空题

1．设二进制数 a 是 00101101，若想通过异或运算 a^b 使 a 的高 4 位取反，低 4 位不变，则二进制数 b 应是_____。

2．以下程序的功能是将 a 数据的低 4 位取反。程序运行后的输出结果是_____。

```
#include <stdio.h>
void main()
{   unsigned char a=0x39,b=0xf;
    a=a^b;
    printf("%x\n",a);
}
```

3．执行下面的程序段后，b 的值为_____。

```
int x=35;
char z='A';
int b;
b=((x&15)&&(z<'a'));
```

4．以下程序的输出结果是_____。

```
#include <stdio.h>
void main()
{   unsigned char a=2,b=4,c=5,d;
    d=a|b;
    d&=c;
    printf("%d\n",d);
}
```

5．有以下程序：

```
#include <stdio.h>
void main()
{   int a=1,b=2,c=3,x;
    x=(a^b)&c;
    printf("%d\n",x);
}
```

程序运行后的输出结果是_____。

第 10 章　文　　件

一、单项选择题

1．语句 FILE *fp;定义了一个指向_____。

A．某一个文件的变量　　B．某一个文件的结构体变量

C．FILE 文件的变量　　D．FILE 类型的指针变量

2．以读写方式打开一个已有的文本文件 file1，下面 fopen 函数正确的调用方式是_____。

A．FILE *fp;
fp=fopen("file1","r");

B．FILE *fp;
fp=fopen("file1","r+");

C．FILE *fp;
fp=fopen("file1","rb");

D．FILE *fp;
fp=fopen("file1","rb+");

3．语句 fp = fopen("file1","r"); 表示_____。

A．打开文件 file1，可从该文件输入数据
B．打开文件 file1，可向该文件输出数据
C．打开文件 file1，可向该文件添加数据
D．打开文件 file1，可向该文件输入/输出数据

4．若 fp 为文件指针，且文件已正确打开，以下语句的输出结果为_____。

```
fseek(fp,0,SEEK_END);     i=ftell(fp);     printf("i=%d\n", i);
```

A．fp 所指文件的记录长度
B．fp 所指文件的长度，以字节为单位
C．fp 所指文件的长度，以比特为单位
D．fp 所指文件当前位置，以字节为单位

5．如果希望向文本文件末尾添加数据,则应以_____方式打开文件。

A．"r"　　B．"w"　　C．"a"　　D．"ab"

6．关闭文件函数 fclose(fp)的作用是_____。

A．不再为 fp 指向的文件分配内存空间
B．将 fp 指向的文件存入磁盘
C．释放指定文件所占据的内存空间和文件指针
D．将指定文件封闭在某一内存区域

7．有以下程序（提示：程序中 fseek(fp,-2L*sizeof(int),SEEK_END);语句的作用是使位置指针从文件尾向前移 2*sizeof(int)字节）

```
#include <stdio.h>
void main()
{   FILE *fp;  int i,a[4]={1,2,3,4},b;
    fp=fopen("data.dat","wb");
    for(i=0;i<4;i++)  fwrite(&a[i],sizeof(int),1,fp);
    fclose(fp);
    fp=fopen("data.dat","rb");
    fseek(fp,-2L*sizeof(int).SEEK_END);
    fread(&b,sizeof(int),1,fp);/*从文件中读取sizeof(int)字节的 数据到变量b中*/
    fclose(fp);
    printf("%d\n",b);
}
```

执行后输出结果是_____。

A．2　　B．1　　C．4　　D．3

8．以下与函数 fseek(fp,0L,SEEK_SET)有相同作用的是_____。

A．feof(fp)　　B．ftell(fp)　　C．fgetc(fp)　　D．rewind(fp)

9．读取二进制文件的函数调用形式为 fread(buffer,size,count,fp);，其中 buffer 代表的是_____。

A．一个文件指针，指向待读取的文件

B．一个整型变量，代表待读取的数据的字节数

C．一个内存块的首地址，代表读入数据存放的地址

D．一个内存块的字节数

10．整块输入函数 fread(&array,2,16,fp)的功能是_____。

A．从数组 array 中读取 16 次 2 字节数据存储到 fp 所指的文件中

B．从 fp 所指的数据文件中读取 16 次 2 字节的数据存储到数组 array 中

C．从数组 array 中读取 2 次 16 字节数据存储到 fp 所指的文件中

D．从 fp 所指的数据文件中读取 2 次 16 字节的数据存储到数组 array 中

11．有以下程序：

```
#include <stdio.h>
void WriteStr(char *fn,char *str)
{   FILE *fp;
    fp=fopen(fn,"w");fputs(str,fp);fclose(fp);
}
void main()
{   WriteStr("t1.dat","start");
    WriteStr("t1.dat","end");
}
```

程序运行后，文件 t1.dat 中的内容是_____。

A．Start　　B．end　　C．startend　　D．endrt

12．有以下程序：

```
#include <stdio.h>
void main()
{   FILE *fp; int a[10]={1,2,3,0,0},i;
    fp=fopen("d2.dat","wb");
    fwrite(a,sizeof(int),5,fp);
    fwrite(a,sizeof(int),5,fp);
    fclose(fp);
    fp=fopen("d2.dat","rb");
    fread(a,sizeof(int),10,fp);
    fclose(fp);
    for(i=0;i<10;i++)
    printf("%d,",a[i]);
}
```

程序的运行结果是_____。

A．1, 2, 3, 0, 0, 0, 0, 0, 0, 0,　　B．1, 2, 3, 1, 2, 3, 0, 0, 0, 0,

C．123, 0, 0, 0, 0, 123, 0, 0, 0, 0,　　D．1, 2, 3, 0, 0, 1, 2, 3, 0, 0,

13．标准输入文件由系统分配为_____。

A．键盘　　B．显示器　　C．打印机　　D．鼠标

14．下面列出四组函数名，其中_____组为标准文件的输出函数。

A．getchar( ),gets( ),scanf( )　　B．putchar( ),puts( ),printf( )

B．fgetc( ),fgets( ),fscanf( )　　D．fputc( ),fputs( ),fprintf( )

15．fgets(str, n, fp)函数从文件中读入一个字符串，以下正确的叙述是_____。

A．字符串读入后不会自动加入'\0'

B．fp 是 FILE 类型的指针

C．fgets 函数将文件中最多读入 n–1 个字符

D．fgets 函数将文件中最多读入 n 个字符

16．文件是以_____为结束符的。

A．NULL　　B．EOF　　C．$　　D．\n

17．若对文件 fp 操作出错，则函数 ferror(fp)的返回值为_____。

A．0　　B．–1　　C．1　　D．非零

18．完成将文件指针 fp 重新指向 C 文件的开头位置的函数是_____。

A．fseek(fp)　　B．ftell(fp)　　C．rewind(fp)　　D．feof(fp)

19. 若 fp 是指向某文件的指针，且已读到文件末尾，则库函数 feof（fp）的返回值是_____。

A．EOF　　B．–1　　C．非零值　　D．NULL

20．以下程序企图把从终端输入的字符输出到名为 abc.txt 的文件中，直到从终端读入字符#号时结束输入和输出操作，但程序有错。出错的原因是_____

```
#include <stdio.h>
void main()
{   FILE *fout;
    char ch;
    fout=fopen( 'abc.txt','w');
    ch=fgetc(stdin);
    while(ch!='#')
    {   fputc(ch,fout);
        ch=fgetc(stdin);
    }
    fclose(fout);
}
```

A．函数 fopen 调用形式有误　　B．输入文件没有关闭

C．函数 fgetc 调用形式有误　　D．文件指针 stdin 没有定义

二、填空题

1．若 fp 已确定为一个文件指针，d1.dat 为二进制文件，请填空，以便为“读”而打开此文件：fp=fopen(_____);。

2．有以下程序：

```
#include <stdio.h>
void main()
{   FILE *fp1, *fp2;
    fp1=fopen("test1","r");
    fp2=fopen("test2","w");
```

```
    while(!feof(fp1))
        fputc(fgetc(fp1),fp2);
    fclose(fp1);
    fclose(fp2);
}
```

该程序的功能是_____。

3. 下面的程序执行后，文件 test.t 中的内容是_____。

```
#include <stdio.h>
#include<string. h>
void  fun（char *fname，char *st）
{   FILE *myf;
    int  i;
    myf=fopen（fname，"w"）;
    for（i=0;i<strlen（st）;i++）
        fputc（st[i]，myf）;
    fclose（myf）;
}
void  main（）
{   fun（"test.t"，"new world"）;
    fun（"test.t"，"hello，"）;
}
```

4. 有以下程序：

```
#include <stdio.h>
void main()
{   FILE *fp;  int i,k=0,n=0;
    fp=fopen("d1.dat","w");
    for(i=1;i<4;i++)
        fprintf(fp,"%d",i);
    fclose(fp);
    fp=fopen("d1.dat","r");
    fscanf(fp,"%d%d",&k,&n);
    printf("%d %d\n",k,n);
    fclose(fp);
}
```

程序运行后的输出结果是_____。

5. 有以下程序：

```
#include <stdio.h>
void main()
{   FILE  *fp;    int  i=20,j=30,k,n;
    fp=fopen("d1.dat","w");
    fprintf(fp,"%d\n",i);fprintf(fp,"%d\n",j);
    fclose(fp);
    fp=fopen("d1.dat", "r");
    fscanf(fp,"%d%d",&k,&n);    printf("%d  %d\n",k,n);
    fclose(fp);
}
```

程序运行后的输出结果是_____。

6. 下面的程序用来统计文件中字符的个数，请填空。

```
#include <stdio.h>
void main()
{   FILE *fp;
    long num=0;
    if((fp=fopen("fname.dat","r"))==_____)
        {   printf("Can\'t open file!\n"); exit(0);  }
    while(_____)
        {   fgetc(fp);  _____;   }
    fclose(fp);
    printf("num=%d\n",num);
}
```

7. 下面的程序用来对 b 盘上的文件 txt.dat 逐个字符读出并显示在屏幕上，请填空。

```
#include <stdio.h>
void main ()
{   FILE *fp;int c;
    if((fp=fopen("b:txt.dat",_____))==NULL)
        {   printf("Can\'t open file!\n"); exit(0);  }
    while((_____)!=EOF)  putchar(c);
    _____;
}
```

8. 下面程序把从终端读入的文本（用@作为文本结束标志）输出到一个名为 bi.dat 的新文件中。请填空。

```
#include <sidio.h>
FILE *fp;
void main（）
{   char ch;
    if（fp=fopen（____））==NULL）exit（0）;
    while（（ch=getchar（））!='@'）fputc（ch,fp）;
    fclose（fp）;
}
```

9. 已知学生的记录由学号和学习成绩构成，N 名学生的数据已在主函数中放入结构体数组 s 中。请编写函数 fun，函数的功能是：把高于等于平均分的学生数据放在 b 所指的数组中，高于等于平均分的学生人数通过形参 n 传回，平均分通过函数值返回。请在函数 fun 的花括号中填入你编写的若干语句。

```
#include <stdio.h>
#define  N  12
typedef  struct
{   char  num [10];
    double  s;
}STREC;
double  fun  ( STREC  *a,  STREC  *b,  int  *n )
{ _____ }
```

```
void main ()
{   STREC  s[N]={{"GA05",85}, {"GA03",76}, {"GA02",69}, {"GA04",85},
        {"GA01",91}, {"GA07",72}, {"GA08",64}, {"GA06",87},
        {"GA09",60}, {"GA11",79}, {"GA12",73}, {"GA10",90}};
    STREC  h [N],t; FILE  *out ;
    int  i,j,n;  double  ave;
    ave=fun ( s, h, &n );
    printf ( "The %d student data which is higher than %7.3f:\n", n, ave);
    for ( i=0; i<n; i++ )
        printf  ("%s  %4.1f\n", h[i].num, h[i].s);
    printf ("\n");
    out=fopen("out 12.dat","w");
    fprintf(out,"%d\n%7.3f\n",n,ave);
    for(i=0;i<n-1;i++)
        for(j=i+1;j<n;j++)
            if(h[i].s<h[j].s) {t=h[i]; h[i]=h[j]; h[j]=t;}
    for(i=0;i<n;i++)
        fprintf(out,"%s %4.1f\n",h[i].s);
    fclose(out);
}
```

10．函数 fun 的功能是，将两个两位数的正整数 a、b 合并形成一个整数放在 c 中。合并的方式是：将 a 数的十位和个位数依次放在 c 数的千位和十位上，b 数的十位和个位数依次放在 c 数的百位和个位上。例如，当 a＝45，b＝12。调用该函数后，c＝4152。数据文件 IN1.DAT 中的数据不得修改。请勿改动主函数 main 和其他函数中的任何内容，仅在函数 fun 的花括号中填入编写的若干语句。

```
#include <stdio.h>
void fun(int a, int b, long *c)
{ _____ }
void  main()
{   int a,b; long c;
    clrscr();
    printf("Input a, b:");
    scanf("%d%d", &a, &b);
    fun(a, b, &c);
    printf("The result is: %d\n", c);
    NONO();
}
NONO () /* 本函数用于打开文件，输入数据，调用函数，输出数据，关闭文件*/
{   FILE *rf, *wf ;
    int i, a,b ; long c ;
    rf = fopen("in1.dat", "r") ;
    wf = fopen("bc01.dat","w") ;
    for(i = 0 ; i < 10 ; i++)
    {   fscanf(rf, "%d,%d", &a, &b) ;
        fun(a, b, &c) ;
        fprintf(wf, "a=%d,b=%d,c=%ld\n", a, b, c) ;
    }
```

```
    fclose(rf) ;
    fclose(wf) ;
}
    /* 文本文件 in1.dat 内容如下:
    45,12
    63,54
    14,78
    88,91
    71,13
    12,45
    72,32
    29,99
    34,43
    22,44       */
    /* 正确程序输出的 bc01.dat 内容如下:
    a=45,b=12,c=4152
    a=63,b=54,c=6534
    a=14,b=78,c=1748
    a=88,b=91,c=8981
    a=71,b=13,c=7113
    a=12,b=45,c=1425
    a=72,b=32,c=7322
    a=29,b=99,c=2999
    a=34,b=43,c=3443
    a=22,b=44,c=2424       */
```

第 3 部分　上 机 指 导

上机实验是 C 语言学习的重要环节，也是掌握程序设计技能的必经之路。本部分介绍 C 程序的开发过程和操作环境。有两个重要的开发环境，一个是基于 DOS 操作系统的 Turbo C 集成环境，一个是基于可视化平台的 Visual C++系统。后者应用已日见广泛，2008 年新的全国计算机等级考试大纲已将其作为主要的实验考核环境。本部分将分别介绍在这两种实验环境下，如何进行 C 源程序的输入、编辑、编译、连接、运行、调试等过程，最后介绍常见的上机错误和纠正办法。

第 1 章　概　　述

用高级语言书写的程序称为源程序。C 源程序编写好后，还需要上机进行源程序的输入、编辑、编译、连接、运行、调试等过程。在各个过程中，都有可能出现一些没有估计到或疏忽的错误。因此，程序开发的全过程是在计算机提供的环境下，不断地发现错误，不断地纠正错误，直到产生一个正确的可执行程序为止。

和其他高级语言一样，C 语言程序的开发过程如图3.1所示，一般有下述几个过程。

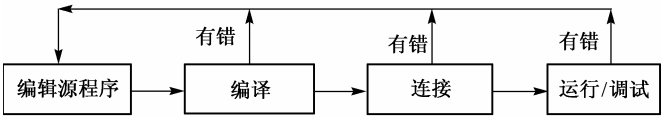

图 3.1　C 程序的开发过程

1. 确定算法和编码

算法包括确定数据结构和设计算法。根据要完成的任务和指定的输入数据与输出结果确定存放数据的数据结构，由粗到精，用流程图描述出详细的算法。

编码，即编写 C 源程序。根据算法描述或者程序流程图，严格按照 C 语言的语法规定编写程序，可书面编写或直接利用编辑程序上机输入。

编好的程序能否可行，必须通过上机调试。它包括源程序的编辑、编译、连接、运行等过程。

2. C源程序的输入和编辑

编辑就是在编辑程序的环境下，将编写好后的 C 源程序输入计算机，并对其进行修改，最后以源文件的形式保存在外存储器上的过程。通常任何能进行 ASCII 码文本编辑的软件，如 Windows 系统中的记事本、DOS 系统中的 EDIT 等都能完成 C 源文件的输入、编辑和存盘功能。

在编辑过程中可能会产生这样或那样的错误。除了算法本身的问题及源程序本身书写的错误外，常常是在程序输入过程中出现错字、漏字或多输入字符的情况。一般编辑程序都具有相应的增、删、改操作，以及数据块的复制、移动、删除等功能或命令。一旦有错，先进

行光标定位，然后修改。最好养成以标准的缩格方式输入源程序的习惯，必要时写些注释，这样程序的可读性高，便于以后修改。源程序改好后，要进行存盘操作，并给源文件取名。C 源文件的名字由用户定，但必须有后缀“.c”，如 test.c。存盘后即可进入编译阶段。

3. C源程序的编译

编译是利用编译程序（编译器），把源程序文件翻译成相应的用机器语言描述的程序文件，即目标文件的过程。

编译一般有以下几个阶段：

(1) 词法分析

对读入的源程序进行扫描，识别出其中具有独立意义的语法单位，即单词符号。若遇到预处理命令，则启动编译程序中的预处理模块。对于#include，搜索指定的包含文件或者库文件，插入到预处理命令处；对于#define，进行相应的宏展开，再进行词法检查。

(2) 语法分析

再次对源程序进行扫描，根据语法规则对单词符号串进行语法分析，识别出不同的语句、表达式和程序的逻辑结构。

在词法和语法分析过程中，一旦发现错误，如有不合要求的单词符号，不合语法的语句等，就要在屏幕上指出错误的位置、错误的性质（出错码），给出相应的出错信号与提示，并且终止编译过程，等待用户再次使用编辑程序进行查找和纠正错误。要注意的是，有时一处的错误可能带来多处出错，比如某一变量定义不当，那么源程序中所有出现这个变量符号的地方，编译程序将不能识别，而全部给出出错信息。这时只要排除掉一个错误，可能会消除多个错误。

如果语法识别没有问题，则编译程序会根据不同的句型和结构，转去执行相应的语义处理程序，从而产生相应的中间语言程序。

(3) 优化中间程序

对二次扫描产生的中间语言程序进行优化处理，以提高程序的执行效率。

(4) 分配存储空间

对程序中的各种类型的常量、变量及数组等构造型数据分配相应的存储空间。

(5) 生成目标程序

将优化的中间语言程序转换为用特定的机器语言描述的目标程序，并存储到磁盘文件中。目标文件带有规定的后缀，在 Windows 和 MS-DOS 下为.obj，在 UNIX 下为.o，在 CP/M 下为.CRL 等。

4. 连接目标文件

编译后的目标文件是浮动的程序模块，不能直接运行，必须使用连接程序，把它和其他目标程序模块及系统提供的标准函数库等进行连接，才能够产生可运行的可执行文件。

在连接过程中，也可能出现错误。这些错误，有的在于各个目标模块之间的不协调，有的可能是引用了未定义的目标模块等。有些错误（如警告错误）并不影响可执行程序的产生。有些致命性的错误，会终止连接过程，这时就要返回到编辑程序，对源程序进行相关修改，然后再次进行编译和连接。

连接成功后产生的可执行程序的文件名后缀，在 Windows 和 MS-DOS 系统下为.exe，在 UNIX 系统下为.out，在 CP/M 系统下为.com。

5. 程序的执行和调试

在操作系统下，可直接运行可执行程序。如果程序运行中出现错误，或者运行结果不符合要求，还必须进行程序的调试，即检查源程序甚至审查算法。在排除错误后，再进行源程序的编辑、编译、连接与执行，直至得到正确结果。

程序调试最好利用系统提供的调试工具。调试工具是一种实用程序，通过它可以直观动态地跟踪展现被调试程序的执行流程，并控制程序的执行。常用的跟踪手段主要有程序的单步执行和断点检查。

程序的单步执行指每执行完一条指令或语句，使程序暂停；断点检查指程序执行前设置一个或多个程序断点，使程序运行到断点处暂停。

程序暂停时，用户可检查程序此时状态，如有关变量的数据情况等。还可设置一个或多个观察点，即确定一些观察量，如某些变量。程序每运行到观察点，或者一旦观察量的值发生变化，程序就会暂停，并显示改变该变量的相关语句，统计改变的次数等信息。调试工具给用户查错、排错带来很大方便。我们应有意识地掌握它。因为谁也不能保证，编程一次就能通过，特别是稍稍复杂的程序，反复修改调试是很平常的。

6. 整理并写出文档资料

程序能正确运行，一般认为程序设计已经完成，接下来的工作是整理相关的文档资料。这些资料包括程序的功能、设计者、使用环境和操作说明等。这些说明文字，可以以注释语句的形式列入程序模块中，也可以作为文本文件的形式予以保存。当然，若在源程序中插入注释语句，最好重新编译、连接，甚至执行一次，以防实质性的变动。程序说明资料虽然不是程序设计的必要阶段，但是对于规模较大，程序模块较多的程序，附有有关说明资料很有必要，它有助于今后对源程序的阅读、维护和修改。

第 2 章　Turbo C程序开发实践

Turbo C 2.0 是目前微机上应用最广泛的 C 编译系统之一，它由美国 Borland 公司开发，基于 DOS 平台，具有友好的用户界面和丰富的库函数，为用户开发 C 程序提供了一个从编辑、编译、连接、调试到执行的方便的集成开发环境。

2.1　Turbo C的集成开发环境

2.1.1 Turbo C的安装

Turbo C 需要在 PC 系列计算机上的 DOS 操作系统下运行。今天用到的 PC 计算机都具备该运行环境。现代 Windows 操作系统上就有“命令提示符”窗口，该窗口提供了模拟的 DOS 操作系统环境。

Turbo C 系统提供了一个安装程序 install.exe。安装时要注意安排有关文件的路径和文件夹。一般将 install.exe 所在的文件夹作为当前目录，当双击 install.exe 运行安装程序时，程序会要求用户根据屏幕的提示指定存放 Turbo C 系统文件的目录和存储模式。一般用户可采用系统默认的方案。例如，如果 install.exe 在 D:\下，则安装后，会在 D:\下建立 TC 子目录。为了以后叙述方便，本书假定 Turbo C 系统所在路径为 C:\TC，在 TC 文件夹中应包含下述主要文件：

① 在 C:\TC 文件夹中，有 tc.exe、tcc.exe、make.exe 等可执行文件，并有 INCLUDE、LIB 等两个下级子文件夹；

② 在 C:\TC\INCLUDE 子文件夹中，有 stdio.h、string.h、math.h 等头文件；

③ 在 C:\TC\LIB 子文件夹中，有 maths.lib、mathl.lib、graphics.lib 等库函数文件。

其中，tc.exe 是一个集编辑、编译、连接、运行及调试为一体的基本软件，它为用户开发 C 程序提供了一个从编辑到调试的方便的集成开发环境。tcc.exe 的功能类似于 UNIX 系统中的 CC 命令，是一个传统方式上的编译程序，它用于 DOS 环境，可对.c 源程序进行编译，产生.obj 程序，再利用 DOS 系统提供的 LINK 程序进行连接，产生.exe 可执行程序。本书主要介绍 tc.exe 的应用。TCC 的使用请读者参阅有关书籍。

有关头文件和库函数文件的应用，主教材中已有介绍，本书不再赘述。

2.1.2 TC的启动和TC集成环境

1. TC的启动

(1) 进入“命令提示符”窗口

在 Windows 平台上，有三种方式可进入“命令提示符”窗口。

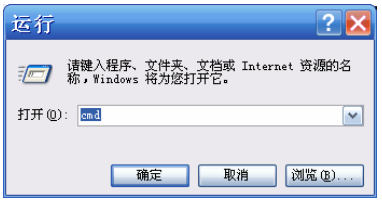

图 3.2 Windows 系统的“运行”对话框

第 1 种方式：按照“开始→程序→附件→命令提示符”顺序打开“命令提示符”窗口。

第 2 种方式：按照“开始→运行”顺序打开“运行”对话框（如图3.2 所示）。在对话框中输入 cmd，单击“确定”按钮，即进入“命令提示符”窗口。

第 3 种方式：直接双击 DOS 环境下开发的可执行程序（.exe 或.com 程序），此时系统进入“命令提示符”窗口并执行相关程序。

小窍门：建议用第 2 种方式，因为第 1 次输入 cmd 后，以后会保留原来的输入，这样再次进入就只需单击“确定”按钮，比第 1 种方式简便些。

“命令提示符”窗口可扩展为整个屏幕。这一操作可用 Alt+Enter 键盘操作实现。若要从全屏幕窗口恢复为桌面上的窗口，也可用 Alt+Enter 键盘操作来实现转换。

(2) 进入 C:\TC 文件夹

在“命令提示符”窗口中输入 cd\tc 并回车，该命令要求将当前目录改变为\tc。如图3. 3 所示。

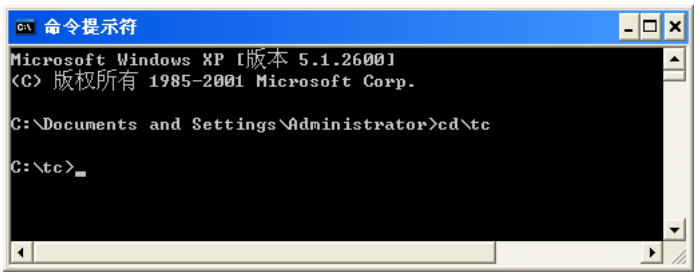

图 3.3 在 Windows 桌面上进入“命令提示符”窗口

(3) 启动 TC

TC 有两种启动方式。一种是直接输入 TC，另一种是输入：

TC　C 源程序文件名

后者不仅进入 TC 集成工作环境，而且将指定的要编辑的 C 源程序文件装入内存，并将文件内容显示在编辑窗口中。

启动 TC 后的界面如图3.4所示。

图3.4 中，中间部分为编辑窗口，其上面一行为主菜单。下拉菜单可进行各种相关操作。最下面一栏是提示常用的快捷操作功能键。

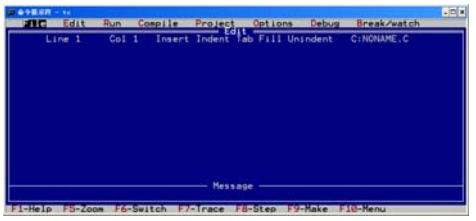

图 3.4　启动 TC 后的界面

2. TC主菜单的操作

主菜单的激活：按 F10 键。

各个菜单的选择：按→键或←键选定后下拉，也可按 Alt+菜单项第 1 个字母来实现。如按 Alt+F 选定 File 菜单。后面将对常用到的菜单功能项分别进行介绍。

3. TC工作环境的设置

Turbo C 在安装时，自动设置了一个内部默认的程序开发环境，即对编辑程序、编译程序、连接程序等的工作状态和环境变量给定了初始值，并保存在一个特定的名为 TCCONFIG.TC 的配置文件中。该配置文件位于 TC 子目录下。TC 启动后，自动将该文件调入内存并遵循其配置。用户在 TC 使用中，很可能由于工作环境与配置文件不相符而使 TC 操作出现错误或不方便操作。如编译时找不到相关包含文件而出错，连接后的可执行程序文件存不到想要存放的地方等，这就需要对配置文件进行修改。但是配置文件不能直接用 TYPE 命令显示内容，也不能用编辑程序进行修改，而只能在 TC 集成环境下进行。具体步骤如下：

(1) 在 TC 环境下按 F10 键激活菜单栏，选 Options 菜单（如图3.5 所示）：

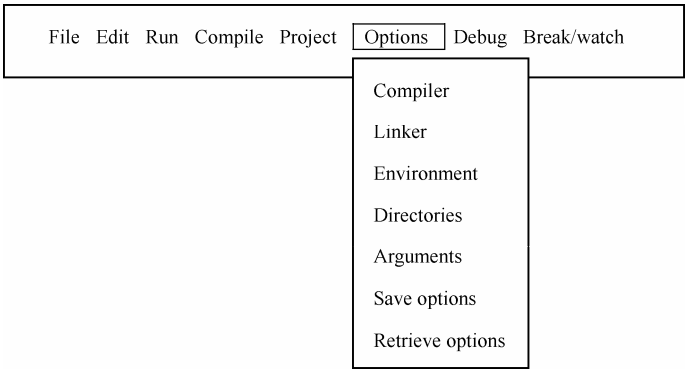

图 3.5　TC 菜单栏和 Options 菜单

(2) 在 Options 菜单中选 Directories 项，出现如图3.6所示的框（中文为作者注释）：

(3) 依次按照用户的要求和系统实际情况设置好各个目录，按 Esc 键返回 Options 菜单。

(4) 选定 Save options 项，系统出现要求存储配置文件 TCCONFIG.TC 的文本框，用户按回车键认可。此时，新的配置就存放于磁盘上，以后 TC 系统就按此配置，而无需再次设置了。

Include directories: C:\TC\INCLUDE	指明包含文件所在的目录
Library directories: C:\TC\LIB	指明库函数等文件所在的目录
Output directories: C:\TC\USER	指明用户建立的存放.exe 和.obj 文件所在的目录
Turbo C directories: C:\TC	指明编译程序所在的目录
Pick file name:	
Current Pick file:	

图 3.6 文件目录设置

注意：① 进行 C 程序操作前，一定要检查或做好 TC 工作环境的设置。因为 TCCONFIG.TC 保存的环境配置不一定与用户当前所处的环境吻合。例如，包含文件的目录原配置为 D:\TC\INCLUDE，而实际位于 C:\TC\INCLUDE 中，这样就会带来找不到包含文件的错误。② 进行环境设置后，一定要选 Save options 项，以便将设置好的信息保存到文件 TCCONFIG.TC 中。如果不做这一项工作，则所进行的设置就白做了。

2.1.3 文件（File）操作

1. 新建文件

选菜单 File 下的 New 项，光标直接停留在编辑窗口左上角，此时编辑窗口上方显示的文件名为 NONAME.c，表示新建文件未取名字，暂时冠以 NONAME.c。

2. 打开文件

选菜单 File 下的 Load 项或按 F3 键。系统会弹出 Load File Name 窗口，显示“*.c”字样，要求用户输入要编辑的文件名。如果用户按回车，则会出现用户目录下的所有 C 源程序文件，供用户选择。用户选定文件后，系统即将指定文件装入内存，并在编辑窗口中显示文件内容。

打开文件的另一方法是选菜单 File 下的 Pick 项或按 Alt+F3，系统会将最近用户编辑的源文件列出供用户选定。

3. 文件保存

选菜单 File 下的 Save 项或按 F2 键。系统会弹出 Rename NONAME 窗口，要求输入文件名替代 NONAME 文件名。也可选菜单 File 下的 Write to 项，即将当前文件内容转存为另一个文件，相当于 Word 中的“另存为”功能。

小窍门：如果要输入的源程序文件格式同当前源程序文件格式基本一样，不妨使用 Write to 项，将当前文件内容转存为要输入的文件，再进行修改，可省去很多输入的繁杂工作。

4. 临时返回操作系统

如果希望进行 DOS 系统下的操作，如查看目录中有哪些文件等。这时可选菜单 File 下的 Os shell 项，机器会暂时回到 DOS 界面，用户可输入有关 DOS 命令进行相关操作，此时若要回到 TC 界面，只需输入命令 exit 即可。

5．退出

选菜单 File 下的 Quit 项或按 Alt + X，机器会退出 TC 而回到 DOS 系统。在退出前，系统可能会询问所编辑的文件是否存盘，若是 NONAME 文件，还要求用户输入新的

文件名。

2.1.4　编辑（Edit）操作

1. Edit编辑窗口简介

Edit 编辑窗口大致由 3 部分组成。

① 最上方一行指示编辑状态，主要有 Line、Col，指示光标当前在整个文本中的位置（行、列号），当光标移动时，行、列号会随之改变。Insert 表示插入状态，由 Insert 键控制。若无 Insert 字样则表示改写状态。这一行的右方显示被编辑的文件名。

② 中间部分为正文区，程序输入即在此进行。

③ 窗口下部为信息（Message）区。编译出错信息、程序调试信息都在此区域内显示。

2. 程序输入

启动 TC 后，即进入编辑状态，可直接输入程序。如果光标不在编辑窗口程序行中，可按 F10 激活主菜单选择 Edit 项就会使光标回到编辑窗口中。

程序输入时请注意几点：

① 注意区分外形相近的字符，比如大写字母“I”、小写字母“l”与数字“1”；字母“o”与数字“0”等。另外，大小写混用情形等都会带来错误。

② 程序代码中不允许用中文标点符号来代替西文标点符号，否则出错。

③ 以标准的缩格方式输入源程序，最好采用按 Tab 键方式实现缩格。Tab 键的跳格列数可以通过 Options 菜单下的 Environment 菜单项中的 Tab size 项进行调整。但要注意，缩格写法并不能让编译程序识别出程序结构，它只是为了用户阅读程序方便。

④ 必要时输入注释，这样程序的可读性高。

⑤ 每个语句后一定要有分号。这是编译程序识别 C 语句的标志。标识符之间哪怕增加若干空格甚至回车，一般都不会影响编译程序对语句的识别。

⑥ 注意{}、[]、()、"、'等符号的配对。

⑦ 可采用块操作来加快程序输入。

3. 光标定位

对程序修改时，应先进行光标定位。除按→←↑↓键移动光标外，还主要有下面的操作：

Home：	光标移至行首
End：	光标移至行尾
Page Up	光标上移一屏
Page Down	光标下移一屏
Ctrl＋Home：	光标移至当前页第一行
Ctrl＋End：	光标移至当前页最后一行
Ctrl＋Page Up：	光标移至源程序第一行
Ctrl＋Page Down：	光标移至源程序最后一行

4. 字符编辑

插入：在插入状态下，输入字符将在光标处插入。

改写：在改写状态下，输入字符将取代光标处的字符。按 Insert 键，可实现插入/改写状态的转换。

删除：按 Delete 键，删除光标处字符。按 Backspace 键，则删除光标左边的字符。按 Ctrl+Y 则删除光标所在的一行。

5. 块操作

所谓块，是指一个连续的字符块。块操作以块为单位进行处理，比单个字符的处理效率高。块操作有两种方式，命令方式和可视化操作方式。

(1) 命令方式

作块标记：按 Ctrl＋KB，在光标处标记块首，按 Ctrl＋KK，在光标处标记块尾。

块复制：将光标移到需要复制的位置，按 Ctrl＋KC 键，此时被标记的块就会复制到光标所在处。

块移动：将光标移到需要移动的目标位置，按 Ctrl＋KV 键，此时被标记的块将搬移到光标处。

块删除：按 Ctrl＋KY，被标记的块将被删除掉。

块取消：取消所作的块标记，按 Ctrl＋KH 即可。

(2) 可视化操作方式

将鼠标停在“命令提示符”窗口的标题栏中，单击右键，在出现的菜单中选择“编辑”，即出现编辑菜单，如图3.7所示。

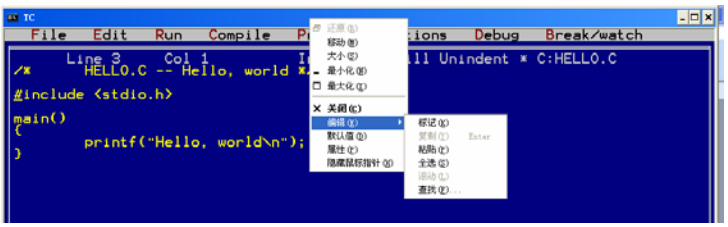

图 3.7 可视化块操作方式

如选定“标记”，则按鼠标移过的字符会标上相应的块标记；“复制”、“粘贴”、“全选”、“查找”等其他操作的含义和过程与一般 Windows 操作相同。

另外，在 Windows 2000 平台上，“命令提示符”窗口被称为 MS DOS 窗口，在其窗口上方的工具栏中具有“标记”、“复制”、“粘贴”等图标按钮，也很容易实现标记文本块与块的复制、移动等操作。

2.1.5 单程序模块的编译、连接和运行操作

1. 单程序模块的上机操作

TC 对程序的编译、连接和运行操作非常方便，它既可以一步完成单个程序的编译、连接和运行操作，又可以对多个程序模块进行组装和连接处理。

(1) 编译、连接、执行一步法操作

当源程序编辑好以后，选主菜单上的 RUN 菜单项下的 Run 项（也可以在全屏幕状态下按 Ctrl + F9），此时 TC 按照编译、连接的顺序在当前目录中或者 Options 菜单指定的输出目录中生成与源程序同名的目标文件和可执行文件，然后自动运行。

如果在编译过程中出错，TC 会自动停下来，并弹出一个窗口，显示错误数目和编译不成功的信息，同时在编辑窗口的 Message 区域显示出错程序行的行号和错误信息。此时用户必须再次编辑程序，再次选定 Run 项，直至通过编译连接，进行程序运行。

在程序运行中，如果需要从键盘输入数据，则系统隐去 TC 界面，出现 DOS 窗口，并让光标在屏幕上闪烁，等待用户输入。

当程序运行完毕，按程序要求显示结果后立即返回 TC 集成环境。由于计算机运行速度很快，用户根本看不到运行结果。怎么办呢？这时，用户可选定 RUN 菜单项下的 User screen 项，则系统隐去 TC 界面，出现 DOS 窗口，就可以观察运行结果了。此时只要随便按一下键，即返回 TC 集成环境。当然，也可选择 File 菜单下的 Os shell 项，进入 DOS 界面来看结果，但不如前者操作简便。

由于执行 Run 项操作，能产生可执行文件，因此也可退出 TC，直接在 MS DOS 环境下运行可执行程序。

(2) 编译、连接一步法操作

编译连接：选 Compiler 菜单下的 Make EXE file 项，一步完成源程序的编译连接。编译中若出错，则显示和处理情况与执行 Run 项相同。编译成功，会生成与源文件同名的.obj 文件。若连接出错，则显示和处理情况也与执行 Run 项相同。若连接成功，则会生成与源文件同名的.exe 文件。

运行：执行 Run 项或退出 TC 在 DOS 下运行.exe 文件。

(3) 分别编译、连接的操作

编译：选 Compiler 菜单下的 Compile to OBJ 项，进行编译。若编译出错，显示和处理与执行 Run 项相同；若编译成功，则会生成与源文件同名的.obj 文件。

连接：选 Compiler 菜单下的 Link EXE file 项，则对已生成的目标文件进行连接。若连接出错，则显示和处理情况与执行 Run 项相同；若连接成功，则会生成与源文件同名的.exe 文件。

运行：执行 Run 项或退出 TC 在 DOS 下运行.exe 文件。

上述三种操作，对单程序模块来说，显然采用编译、连接、执行一步法较为方便。

2. 单模块程序上机实例

本节结合主教材第 1 章例 1.3，具体说明利用 Turbo C 上机操作的过程。例 1.3 如下：

例 1.3　求两数中的大数。

```
main()                          /* main()为主函数，C 程序的执行起点 */
 {  int  a,b,c;                 /* 说明语句，定义 a,b,c 为 int 型变量 */
    int max(int x,int y);       /* 声明本函数要调用 int 型函数 max，调用参数有 2 个，
                                   为 int 型 */
                                /* 以下为执行语句 */
    printf("Input a,b:");       /* 提示输入 a、b，末尾不换行 */
    scanf("%d,%d",&a,&b);       /* 从键盘输入 2 个数据给 a 和 b，数据之间以逗号分隔 */
    c=max(a,b);                 /* 调用函数 max()，将函数值赋给 c */
    printf("max=%d\n",c);       /* 输出 c */
 }
int max(int x,int y)            /* 定义 max()函数，函数值为 int 型，形式参数 x、y
                                   为 int 型 */
  { int  z;                     /* 定义本函数要用到 int 型变量 z */
    if (x>y)
        z=x;
```

```
    else
        z=y;
    return z;                    /* z 的值作为函数值返回调用处 */
  }
```

操作步骤如下：

第 1 步：进入“命令提示符”窗口。

第 2 步：进入 TC。

第 3 步：输入程序。假定输入程序如下（为便于后面分析，在各程序行首加了行号）：

```
1:  main()
2:    { int  a,b,c                  /* 注意句末没有分号 */
3:      int  max(int x,int y);
4:      printf("Input a,b:");
5:      scanf("%d,%d",&a,&b);
6:      c=max(a,b);
7:      printf("max=%d\n",c);
8:    }
9:  int max(int x,int y)
10:   { int  Z;                    /* 注意定义的变量名是大写字母 Z */
11:     if (x>y)
12:         z=x;
13:     else
14:         z=y;
15:     return z;
16:   }
```

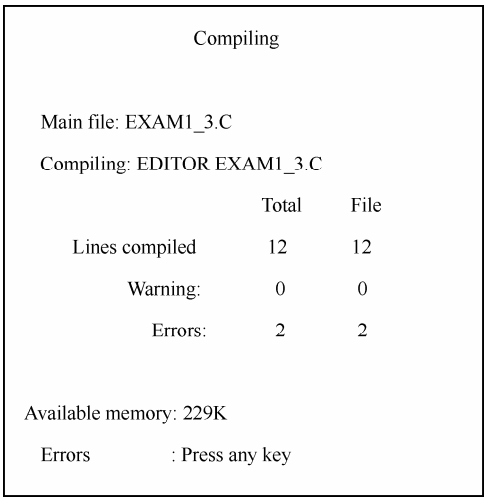

图 3.8 编译出错时的显示

第 4 步：保存程序。按 F10 键，下拉 File 菜单，选 Save 项，在 Rename NONAME 会话框中输入文件名：exam1_3 并回车。（注：文件名可不输入扩展名，系统默认为.c）。此时编辑窗口的右上方文件名变为 EXAM1_3.c。

第 5 步：编译连接执行程序。按 F10 键，下拉 Run 菜单，选 Run 项，此时屏幕上出现如图 3.8所示的 Compiling 文本框。

文本框向用户报告有 2 个错误，提示用户了解情况后按任意键。

第 6 步：程序排错。按任一键，Compiling 框消失，编辑窗口的下方 Message 区域出现：

```
Compining  C:\TC\USER\ EXAM1_3.C:
Error  C:\ TC\USER\EXAM1_3.C  3:   Declaration syntax error in function main
Error  C:\TC\USER\ EXAM1_3.C  12:   Undefined symbol ‘z’ in function max
```

信息框中给出了 2 条出错信息。每一条指出了出错的文件名，出错行号及出错原因。其中，第 1 行是以光带形式加盖，与之对应的是程序中的第 3 行，它也同样以光带加盖：

```
          int max(int x,int y);
```

按↑↓键可改变光带的位置，同时程序行上的光带也会随之移动。例如，在 Message 区域将光带移到第 2 条出错信息，则程序编辑区中的光带也由第 3 行移到第 12 行，即出错信息

与出错程序行是一一对应的。

知道了出错原因和出错程序行，排错似乎很容易。第 1 行出错信息指出第 3 程序行有“在主函数中声明语法错误”，事实上第 3 程序行并没有错，错的是第 2 程序行句末少了个分号。那么，系统为什么说第 2 行有问题呢？原因很简单，由于第 2 行无分号，系统将第 2 行和第 3 行看成一个语句，即

```
int  a,b,c  int max(int x,int y);
```

这显然不合语法。可见，排错时不能单单只看系统指示的程序行，还得联系上下行一起分析。第 2 出错信息行指出第 12 程序行有“在 max 函数中出现未定义的符号‘z’错误”，为什么变量 z 未定义呢？原来在第 10 程序行定义的是大写 Z 作为变量名，因而引起第 12 行的错误。

找到真正的出错地方，接下来就是改正。按回车键或选定 Edit 菜单项使光标回到程序中。移光标至第 2 行，按 End 键移至行末，输入分号。再移光标至第 10 程序行中的大写字母 Z 处，按 Delete 键先删除 Z，然后输入小写 z。

修改好后，接下来再执行 Run→Run 项，此时屏幕的显示如图3.9所示。

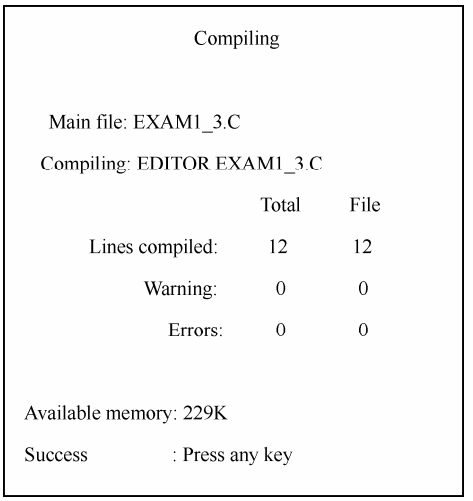

图 3.9　成功编译的窗口显示

“Success”表示编译成功。上述显示可能一闪即过，然后屏幕马上改为 DOS 界面，显示如下：

```
Input a,b: _
```

显然，这是经过编译连接后正在执行的程序。此时输入 5，8 并回车，系统又回到 TC 界面，屏幕上出现源程序编辑窗口，看不到最后结果。

第 7 步：察看程序执行结果。现在选 Run→User screen 项，屏幕改为 DOS 界面，显示如下：

```
Input a,b: 5，8
max=8
```

说明程序得到正确运行。此时任意按一键，又返回 TC 窗口，至此完成该程序的上机实验。

可以调整 Message 窗口使其扩大为整个编辑窗口，方法是当操作点处于 Message 窗口时，按 F5。再次按 F5 可复原。同样，也可以使源程序窗口扩展为整个编辑窗口或复原。

2.1.6　多模块程序的编译、连接和运行操作

1. 多模块程序的上机操作

C 语言的源程序可以划分为一个以上的源文件，即一个程序可以由多个模块文件组成。例如，某一个源程序由 ex1.c、ex2.c 和 ex3.c 组成，那么，这样的源程序在 TC 上该如何编译连接呢？步骤如下：

首先，TC 要求建立一个扩展名为.prj 的项目（Project）文件。这个文件的内容就是组成程序的每个源文件名或者是对每个源文件编译后生成的目标文件名。必要的话，源文件名可包含盘符和路径，扩展名.c 可以省略。例如，我们建立一个名为 ex0.prj 的文件，其内容如下：

```
ex1.c
ex2.c
ex3.c
```

或者内容为

```
ex1.obj
ex2.obj
ex3.obj
```

或者内容为

```
ex1
ex2
ex3
```

其次，TC 要求对要编译连接的项目文件进行登记。登记的方法是在主菜单中 Project 菜单项下选 Project name 项，并在会话窗口中输入项目文件名 ex0.prj。

接下来进行编译连接。对由源文件名组成的项目文件，采用对单模块源程序一样的方法，即对项目文件执行 Run→Run 项或 Compile/Make EXE file 项，生成一个可执行文件 ex0.exe。对由.obj 文件名组成的项目文件，则对项目文件执行 Run→Run 项或 Compile→Link EXE file 项，同样生成一个可执行文件 ex0.exe。

2. 多模块程序上机实例

下面以一个简单的多模块程序为例进行说明。该程序名为 exam0.prj，它由 3 个源文件 exam1.c、exam2.c 和 exam3.c 组成，且只能在一个源文件内含有 main( )函数。程序如下（程序中有意安排了一些错误）：

```
/* 源文件 exam1.c */
extern int max,min;              /* 声明主程序要用到其他模块的变量 */
void main()
{  int a,b;
   printf("Input a,b:");
   scanf("%d,%d",&a,&b);
   emax(a,b);
   printf("max=%d\n",mix);       /* 此处将 max 误写为 mix */
   emin(a,b);
   printf("min=%d\n",min);
}

/* 源文件 exam2.c */
int max;
int emax(int x,int y)
{  if (x>y)
       max=x;
   esle                          /* 此处将 else 写错 */
       max=y;
   return;
}
/* 源文件 exam3.c */
   int min;
   int emin(int x,int y)
   {  if (x<y)
          min=x                  /* 此处漏掉分号 */
```

```
        else
            min=y;
        return;
    }
```

第 1 步：建立 exam0.prj，内容如下;

```
/* 项目文件 exam0.prj */
exam1
exam2
exam3
```

各文件无扩展名，默认为.c，各文件路径默认为当前目录。

第 2 步：项目文件登记并保存。选 Project→Project name 项，并在会话窗口中输入项目文件名 exam0.prj。选 File→Save 项，将要保存的文件命名为 exam0.prj，然后保存在自己指定的目录下。

第 3 步：采用一步法编译连接执行 exam0.prj。选 Run→Run 项，机器按照项目文件给出的源文件的顺序依次编译。显然，exam1.c 未能通过编译，Message 区域首先出现：

```
Compiling c:\tc\user\exam1.c
Error  c:\tc\user\exam1.c 7:  Undefined symbol 'mix'in function main
```

说明出现未定义的符号‘mix’。按回车，屏幕转换为 exam1.c 文件编辑窗口。此时移动光标将“mix”改为“max”。再执行 Run→Run 项，则 exam2.c 未能通过编译，Message 区域出现：

```
Compiling c:\tc\user\exam1.c:
Compiling c:\tc\user\exam2.c:
Error  c:\tc\user\exam2.c 6:  Undefined symbol 'esle'in function emax
Warning  c:\tc\user\exam2.c 6:  Code has no effect in function emax
Error  c:\tc\user\exam2.c 6:  Statement missing ; in function  emax
```

指出在第 6 行有 1 个警告（Warning）和 2 个错误（Error）。错误是：在 emax 函数中有未定义的符号‘esle’和语句丢失了分号。警告信息是，emax 函数中代码不起作用。分析原因是由于系统不能辨识‘esle’，将其看成表达式，而末尾又无分号，因而语句代码可能不起作用。按回车键，屏幕先询问是否保存 exam1.c（因为进行了修改），按 Y 键同意保存，屏幕即转换为 exam2.c 文件编辑窗口。此时移动光标将“esle”改为“else”。再执行 Run/Run 项，则 exam3.c 又未能通过编译，Message 区域出现：

```
Compiling c:\tc\user\exam1.c:
Compiling c:\tc\user\exam2.c:
Compiling c:\tc\user\exam3.c:
Error  c:\tc\user\exam3.c 5:  Statement missing ; in function emin
```

指出在第 5 行有 1 个错误（Error）。错误是：在 emin 函数中有语句丢失了分号。按回车，屏幕先询问是否保存 exam2.c（因为进行了修改），按 Y 键同意保存，屏幕即转换为 exam3.c 文件编辑窗口。此时，移动光标在第 4 行末尾加上分号。再执行 Run→Run 项，回答系统提示保存 exam3.c 后，由于 3 个模块都通过了编译，TC 将 3 个.obj 程序连接后即转入运行。屏幕改为 DOS 界面，显示如下：

```
Input a,b: _
```

输入 5，8 并回车，系统又回到 TC 界面，选 Run/User screen 项，屏幕改为 DOS 界面，显示如下：

```
Input a,b: 5，8
max=8
min=5
```

说明程序得到正确运行。此时任意按一键，又返回 TC 窗口。至此完成了该程序的上机实验，并在 C:\TC\USER 下建立了 1 个名为 exam0.exe 的程序。

上述程序也可以分别编译，再进行连接。

2.1.7 编译、连接操作的控制

在 TC 环境下，通过 Compile、Project 2 个下拉菜单实现对程序的编译和连接。表 3.1 列出了这 2 个菜单中的选择项及其作用。

表 3.1 编译（Compile）和项目（Project）菜单中的选择项及其作用

菜 单	选 择 项	作 用	过 时 检 查
Compile	Compile to OBJ	编译源文件，产生相应目标文件	
	Make EXE file	对.c 文件，编译并连接，产生相应的.obj 和.exe 文件 对.obj 文件，则连接产生.exe 文件	检查
	Link EXE file	连接.obj 文件，产生.exe 文件	不进行
	Build all	同 Make EXE file	不进行
	Primary C file	要求输入编译或 Make 的文件名	
	Get into	显示当前目录、源文件名、文件大小等信息	
Project	Project name	要求输入将要编译连接的项目文件名	
	Break make on	规定终止 Make 的默认条件，有 4 个选择项： Warning：出现警告时停止 Make Error：出现错误时停止 Make Fatal Error：出现致命错误时停止 Make Link：连接前停止 Make，即只生成.obj 文件	
	Auto Dependencies	设置自动依赖关系： on：若.c 和.obj 文件日期不同，则重新编译 off：不进行时间检查	
	Clear Project	清除 Project name 和重置 Message 窗口	
	Remove message	清除 Message 窗口中的出错信息	

几点注意事项：

① TC 编译程序对文件的优先处理顺序是：Project name 中的.prj 文件→Primary C file 中的.c 文件→最后装入编辑窗口中的文件。因此，在执行 Run/Run 项之前，一定要看看 Project→Project name 项或者 Compile→Primary C file 项，否则，有可能编译连接执行的不是当前窗口中的程序。

② 所谓过时检查是指：用 Make EXE file 对.c 文件编译，若.c 文件产生于相应的.obj 文件之前，则拒绝编译，只有更新了.c 文件之后才进行编译。若对.obj 文件进行连接，.obj 文件产生于相应的.exe 文件之前，则拒绝连接，只有更新了.obj 文件之后才进行连接，生成新的.exe 文件。

③ 使用快捷键能加快操作速度。

④ 若要输入汉字，则可启动 Windows 环境下虚拟 DOS 系统中的汉字系统命令：pdos95，或者直接启动希望汉字系统 ucdos。

2.2　Turbo C的动态调试方法

编译程序可以发现程序中的语法错误，但对程序中的逻辑错误检查不出来。另外 C 语言对某些语法检查不很严格，如数组越界问题、数据溢出问题等。当程序运行的结果出现问题时，就要求编程者分析程序，上机对程序进行调试和纠正错误。

TC 集成软件为用户上机调试提供了 1 个源代码级集成调试工具（调试器）。它的操作涉及到 Run、Debug 和 Break/watch 几个菜单。

2.2.1　程序的一般调试

进行程序调试，应先检查 TC 环境是否设置好与调试有关的状态。常用的调试方法有单步调试法和设置断点法，下面分别给以介绍。

1. 上机调试的环境设置

将 Options→Compiler→Code generation→Obj debug information 选择开关置为 On。

将 Debug→Source Debugging 选择开关置为 On。

这样，编译和连接时生成的可执行文件才能包含与调试有关的符号信息。

将 Optiions→Compile→Optimization→Jump optimization 选择开关置为 Off，以便于调试。否则，编译程序优化代码时会引起执行代码行与源代码行的不对应。

2. 单步调试法

所谓单步调试法，又称为程序执行跟踪法。当程序执行时，每执行完一条语句就暂时停下来，这时用户可检查有关变量和表达式的值，了解程序行进的情况，判断是否符合设计要求，然后又运行下一条语句，又进行检查，如此一步一步往下运行，就有可能找到错误在什么地方。单步调试法要用到的功能项如下：

① Run→Trace into（键 F7）项：逐句跟踪程序执行，函数调用时跟踪进入函数内部；

② Run→Step over（键 F8）项：逐句跟踪程序执行，但不跟踪函数调用。

为了在跟踪时观察数据，可使用 Break/watch→Add watch（组合键 Ctrl＋F7）项或者 Debug→Evaluate（组合键 Ctrl＋F4）项，在弹出的 watch 监视窗口中输入要监视的变量或表达式。这样，每次程序单步暂停时，在 watch 窗口里就会看到有关量的当前值。

下面结合 1 个简单的实例说明如何使用单步跟踪法。

设有一个要求两个数的乘积的程序（为了分析方便，加了行号）：

```
1：  main()
2：  {    int x,y,z;
3：       printf("Please input two number:");
4：       scanf("%d,%d",&x,&y);
5：       z=x*y;
6：       printf("z=%d\n",z);
7：  }
```

程序很简单，上机很顺利地通过编译连接，第 1 次运行情况如下：

```
Please input two number: 10，20
z=200
```

程序执行的结果是正确的，但是当第 2 次运行时，结果却是错误的，屏幕显示如下：

```
Please input two number: 300，400
z= -11072
```

为了查错，决定跟踪观察 x、y、z 3 个变量值的变化。为此，选 Break/watch→Add watch 项。在弹出的 watch 窗口中输入 x 并回车。此时，在编辑窗口的下方出现 watch 区域，区域内显示：

- x：Undefined symbol 'x'

由于程序尚未运行，因此系统是辨识不出符号‘x’的。用类似的办法，在 watch 区域内设置 y、z 观察量。接着，开始单步执行程序。按一下 F7 键，发现一个光带罩住第 1 行[内容为 main( )]，表明开始单步运行。再按一下 F7 键，发现光带跳过第 2 行（因为此行是说明语句，是非执行语句），罩住第 3 行[内容为 printf(“Please input two number:”);]。此时，watch 区域显示如下：

- z : 580
  y : 26899
  x : 506

由于程序没给 x、y、z 初值，因此此处 x、y、z 值是任意的。再次按 F7 键，光带移到第 4 行，再按 F7 键，即执行语句 scanf("%d,%d",&x,&y);，此时屏幕转为 DOS 界面，显示如下：

```
Please input two number:
```

输入 300,400 回车，又回到 TC 环境，光带移到第 5 行：z=x*y;。此时，watch 区域显示如下：

- z : 580
  y : 400
  x : 300

这说明变量 x、y 正确地接受了数据。再次按 F7，程序执行语句 z=x*y;，光带移到第 6 行，此时，watch 区域显示如下：

- z : -11072
  y : 400
  x : 300

这说明问题出在语句 z=x*y;上。为什么 z 值不等于 120 000 呢？原因是变量 z 是 int 型，120 000 超出了表数范围。为此，修改语句 int x,y,z;为

```
int x,y; long z;
```

编译连接后，再次运行，结果仍然是错的。再来单步运行调试，还是语句 z = x*y;有错。难道是乘法有错？于是，选 Break/watch→Add watch 项。在弹出的 watch 窗口中输入 x*y 并回车，将表达式 x*y 作为观察对象。当单步运行语句 z=x*y;后，watch 区域显示如下：

- x*y: -11072
  z: -11072
  y: 400
  x: 300

这说明错在 x*y 上。根据同类型的数据进行算术运算，结果仍为同类型的数据这一规则，可知 x*y 的积必是 int 型，表示不了 120 000。于是，修改语句为

```
long  x,y,z;
```

编译连接后，再次运行，显示结果仍然是：z=–11072。错在哪儿？经单步检查，发现错在第 4 行上，x、y 接收的值为 x=–2 096 852，y=1 793 917 328。原因是对 long 型，输入格式符应为%ld。改第 4 行为 scanf("%ld,%ld",&x,&y);，再来运行，结果还是

```
z=-11072
```

单步运行，发现 x、y、z、x*y 值都是对的，问题出在输出语句上。经分析，是格式符%d 与 long 型变量 z 不匹配的缘故。将输出语句改为

```
printf("z=%ld\n",z);
```

改后再次运行，结果完全正确。

如果要对观察量进行增、删、改等编辑操作，可选 Break/watch→Add watch 项（组合键 Ctrl+F7）在 watch 窗口中添加要监视的表达式；选 Break/watch→Delete watch 项可删除 watch 窗口中最后加入的监视表达式；选 Break/watch→Edit watch 项可编辑最后加入的监视表达式。若要删除 watch 窗口中的所有表达式，可选 Break/watch→Remove all watches 项。

3. 设置断点法

单步调试法虽然能跟踪程序执行，但一步一停，速度太慢，特别是当调试较大的程序、循环次数较多的程序时，时间花费太大。如果只希望在有怀疑的语句处了解程序执行的状态，可以采用设置断点法来调试程序。所谓断点，是指要求程序运行时暂停的位置。程序运行前先设好断点。启动运行后，程序会自动地在断点处停下，待用户检查后还可继续运行，直到碰上另一断点或到程序末为止。

TC 调试器可以使用 Break/watch→Toggle breakpoint（组合健 Ctrl + F8）项在任意可执行的语句行上指定为断点，被设置的断点行以高亮度表示。例如，上述被调试程序，为了了解第 5 行执行的结果，可将光标移到第 6 行，选 Break/watch→Toggle breakpoint 项，此时第 6 行被光带罩住，做好了断点标记。同样地，将第 7 行设为断点，以便观察第 6 行执行后的情况。设好断点后，还要在 watch 窗口设好观看数据 x、y、z、x*y。然后选 Run→Run 项启动运行，机器即停在第 6 行断点处，在 watch 窗口会有相应的观察数据。若要继续运行，再次选 Run→Run 项，机器会停在下一断点处（第 7 行），如此直至程序结束。当然，若发现出错，如第 5 行问题，则用户可修改源程序，假定将第 2 行改为“long x,y,z;”，改后又可选 Run→Run 项重新运行，机器又会停在第 1 个断点处。用户可执行前述操作，最后将程序调试好。

如果要消除断点，则可选 Break/watch→Clear all breakpoints 项。

2.2.2　控制程序运行调试的菜单

控制程序运行调试的菜单有 Run 子菜单、Debug 子菜单和 Break/watch 子菜单，下面对各选择项及其功能给以简要的介绍。

1. Run子菜单

Run 子菜单下属的选择项有：

① Run（组合键 Ctrl+F9）

编译、连接生成目标文件和可执行文件并运行。若 Debug→Source debug 为 On，则遇到断点暂停。

② Program reset（组合键 Ctrl+F2）

用于动态调试，终止当前调试工作，释放分配给程序的空间，关闭已打开的文件，但不改变断点设置。

③ Go to cursor（组合键 F4）

用于动态调试，使程序从光带罩住的程序行开始运行到编辑窗口中的光标所在行上。若光标所在行不含可执行语句，则给出警告。

④ Trace into（组合键 F7）

用于动态调试，单步运行。当遇到低一级函数调用时，若 Options→Compiler→Code generation→Obj debug information 选择开关状态为 On，则跟踪进入函数内部。

⑤ Step over（组合键 F8）

用于动态调试，单步运行，但不跟踪函数调用。

⑥ User screen（组合键 Alt+F5）

显示 DOS 操作系统的屏幕输出，按任意键返回编辑窗口。

2. Debug子菜单

Debug 子菜单控制调试器下属的选择项有：

① Evaluate（组合键 Ctrl+F4）

弹出会话窗口，窗口中有三项内容：要计算结果的表达式（Evaluate），表达式的结果（Result）和赋给表达式的新值（New value）。在调试中，可利用该项改变赋给某些量的值。

② Call stack（组合键 Ctrl+F3）

弹出显示函数调用栈的窗口，显示正在运行的程序中函数调用序列。通常，主函数 main()在栈底，正在运行的函数在栈顶。本选择项对了解函数的嵌套层次很有意义。

③ Find function

在编辑窗口中，将光标定位于指定的函数的定义处，该选择项会要求输入要显示的函数名。

④ Refresh display

若编辑屏幕被重写，则该选择项会恢复当前屏幕内容。

⑤ Display swapping

弹出会话窗口，选择编辑屏和用户屏之间转换的三种方式：

- Smart（默认方式）：执行输出代码时显示用户屏，然后返回编辑屏；
- None：不进行屏幕转换；
- Always：每执行一条语句变换一次屏幕。

⑥ Source debugging

弹出会话窗口，选择是否在可执行文件中加入调试信息，有三种方式：

- On：加入调试信息，为源代码级调试作准备；
- Standalone：加入调试信息，但只能使用独立调试工具 Turbo debugger；
- None：不加入调试信息。

3. Break/watch子菜单

下属的选择项有：

① Add watch（组合键 Ctrl+F7）

在 watch 窗口中添加要监视的表达式。

② Delete watch

删除 watch 窗口中最后加入的监视表达式。

③ Edit watch

编辑最后加入的监视表达式。

④ Remove all watches

删除 watch 窗口中的所有表达式。

⑤ Toggle Breakpoint（组合键 Ctrl+F8）

在光标所在行设置断点，再次执行本选择项时清除当前行已设置的断点。

⑥ Clear all Breakpoints

清除所有断点。

⑦ View next Breakpoint

把光标移到下一个断点处。

2.3　TC的Options菜单和常用操作快捷键

2.3.1　Options菜单

Options 菜单控制着 TC 集成环境的状态，可实现对编译、连接及调试的工作方式的设置。下面列出了 Options 菜单的各个选择项及其功能。

1．Compiler选择项

Compiler 选择项的作用是进行编译器设置，子选择项有：

(1) Model：允许用户选择 Tiny、Small（默认）、Medium、Compact、Large、Huge 6 种不同规模存储模式中的某一种。

(2) Defines：可在弹出的窗口内输入宏定义。

(3) Code generation：选择控制生成目标代码的形式，可供选择的有：

① Calling convention：选择 C 或 Pascal 方式传递参数，默认为前者。

② Instruction set：选择产生 8088/8086 或 80186/80286 指令，默认为前者。

③ Float point：选择浮点运算，有 3 种选择：

- Emulation：软件模拟浮点运算（默认）；
- 8087/80287：8087/80287 浮点运算；
- None：无浮点运算。

④ Default char type：规定 char 类型，有 2 种选择：

- Singed：有符号的（默认设置）；
- Unsinged：无符号的。

⑤ Aligement：规定地址对准原则，有 2 种选择：

- Word：字对齐；
- Byte：字节对齐（默认）。

⑥ Generate underbars：规定外部标志：

- On：外部标志加 1 个下画线（默认）；
- Off：不加下画线。

⑦ Merge duplicate strings：合并字符串：

- On：合并匹配的重复字符串，生成规模小一点的程序；
- Off：不合并重复字符串（默认设置）。

⑧ Standard stack frame：控制是否生成更短更快的代码：

- On：不生成更短更快的代码，调试源文件时应置成该方式；
- Off：生成更短更快的代码，不能使用源代码级调试器。

⑨ Test stack overflow：检查堆栈溢出：

- On：产生一段运行时堆栈溢出的代码，因此程序会变得长一些；
- Off：不产生堆栈溢出代码（默认设置）。

⑩ Line numbers：控制在映射文件中是否产生行号：

- On：在符号调试器使用的映射文件中产生行号；
- Off：不产生行号（默认）。

⑪ OBJ debug information：控制在目标文件中是否加入调试信息：

- On：在 obj 文件中加入调试信息（默认）；
- Off：在 obj 文件中不加入调试信息。

(4) Optimization：控制代码生成策略，有下列选择：

① Optimize for：采用优化策略的选择：

- size：生成规模较小的执行代码（默认）；
- speed：生成速度较快的代码。

② Use register variables：是否允许使用寄存器变量：

- On：允许使用（默认）；
- Off：不允许使用。

③ Register optimization：寄存器优化：

- On：尽可能用寄存器保存前面的内容，减少 Load 操作；
- Off：不做以上工作（默认）。

④ Jump optimization：跳转优化：

- On：通过去掉多余的跳转、调整循环及开关语句来压缩代码，加快速度，但不利于调试；
- Off：不做以上工作（默认）。

(5) Source：控制编译器如何处理源代码，有下列选择：

① Identifier length：说明标识符中有效字符个数，可选 1～32，默认为 32。

② Nested comments：是否允许嵌套注释：

- On：允许
- Off：不允许（默认）

③ ANSI keywords only：是否只用 ANSI C 的关键字，不使用 Tuerbo C 的关键字：

- On：是，Tuerbo C 的关键字看着是用户定义的标识符；
- Off：允许使用 Tuerbo C 的关键字（默认）。

(6) Errors：控制编译器处理和响应诊断信息，有下列选择：

① Errors stop after：指定在发现错误时停止编译的错误个数，可选 0～255，默认为 25。

② Warning stop after：指定在发现警告时停止编译的警告个数，可选 0～255，默认为 100。

③ Display warnings：显示警告的类型：

- On：允许显示下列各种类型的警告（默认）：

  Portability warning：移植警告；
  ANSI violations：侵犯了 ANSI 关键字；
  Common errors：常见错误；
  Less common errors：少见错误。

- Off：不显示上述类型的警告。

(7) Names：用于改变代码、数据和 BSS 段的名字，有 3 种选择（一般不用）：

① Code names：代码名；

② Data names：数据名；

③ BSS names：BSS 名。

上述每一种都有 3 种选择：Segment name（段名）、Group name（组名）、Class name（类名）。

2. Linker选择项

Linker 选择项的作用是进行连接器设置，子选择项有：

(1) Map file：是否产生映射文件：

- On：在当前目录或指定的输出目录中产生.map 文件；
- Off：不产生映射文件（默认）。

(2) Initialize segments：连接器是否对未初始化过的段初始化：

- On：对未初始化过的段初始化（一般不需要）；
- Off：不初始化（默认）。

(3) Default libraries：当连接由其他编译器产生的目标文件时，那些编译器可能在目标文件中放入了一个默认库表。本选择项控制连接器是否在那些库中查找所需函数。

- On：查找；
- Off：不查找（默认）。

(4) Graphics library：是否打开查找图形库的开关：

- On：打开图形库（默认，当程序有图形输出时使用）；
- Off：不打开图形库，但在.prj 文件中写上 graphics.lib 可输出图形。

(5) Warn dublicate symbols：检查.obj 文件和.lib 文件中的重复符号：

- On：当.obj 文件和.lib 文件中有重复符号时发出警告信息；
- Off：不检查重复信号（默认）。

(6) Stack warning：是否产生 No stack 警告信息：

- On：产生 No stack 警告信息（默认，在小模式下可能有此警告）；
- Off：不产生 No stack 警告信息。

(7) Case-sensitive link：是否区分大小写字母：

- On：区分大小写字母（C 语言的默认方式）；
- Off ：不区分大小写字母。

3. Environment选择项

Environment 选择项的作用是进行程序编辑环境设置，子选择项有：

(1) Message tracking：当滚动 Message 窗口中的信息时，控制是否跟踪编辑器里的语法错误及跟踪方式，可选择：

- Current file：只跟踪编辑器里的文件（默认）；
- All files：加载、跟踪与错误信息相对应的每个文件；
- Off：不跟踪。

(2) Keep messages：在编译、Make 之前是否清除 Message 窗口中的信息。

- No：保存在当前窗口中的出错信息，再次编译产生的出错信息附在后面（默认）；
- Yes：在编译、Make 之前清除 Message 窗口中的所有信息。

(3) Config auto save：是否保存选择项配置。

- On：当选择 Run→Run 或 File→Os shell 时，只要有选择项被改变，TC 自动地将现行的选择项保存到配置文件中；
- Off：不保存现行选择项（默认）。

(4) Edit auto save：是否将正在编辑的文件自动存盘。

- On：当选择 Run/Run 或 File/Os shell 或 File/Quit 时，TC 自动地将正在编辑的文件存盘；
- Off：不自动存盘（默认）。

(5) Backup files：控制是否产生备份文件：

- On：文件存盘时，将原来的文件以.bak 为扩展名作为备份保存，然后将当前文件存盘（默认）；
- Off：不产生备份文件。

(6) Tab size：选择 Tab 键的跳格数，可选 2～16，默认为 8。

(7) Zoomed windows：把现行激活窗口放大到整个屏幕，类似于快捷键 F5，默认为 off。

(8) Screen size：选择显示屏幕的行数。标准值为 25 行，EGA 为 43 行，VGA 为 50 行，选 Off 为不区分大小写字母。

4. Directories选择项

Directories 选择项的作用是确定头文件、库文件、编译器等所在目录及用户文件目录。

(1) Iclude directories：确定头文件所在的目录。

(2) Library directories：确定库文件所在的目录。

(3) Output directory：确定输出文件所在的目录。

(4) Turbo C directory：确定 TC 所在的目录。

(5) Pick file name：由用户指定 pick 文件名。默认文件名为 TCPICK.TP，在 TC 初始启动时自动加载。若用户没有指定 pick 文件名，则 Options→Directories→Current pick file 项的设置为空；当规定了 pick 文件名时，显示 pick 文件名。

(6) Current pick file：显示当前 pick 文件及其所在目录。

5. Arguments选择项

Arguments 选择项允许用户在集成环境下运行程序时使用参数。例如，编写了 1 个复制文件的程序，它的 main()函数要求带参数运行。那么，在 TC 环境下该如何操作呢？具体作法是，在该程序的编辑环境下，选 Options→Arguments 项，在弹出的命令行窗口中输入程序运行所需的参数，如要复制的源文件名和目的文件名，如图3.10所示，参数之间以空格分隔，然后选 Run/Run 执行程序。

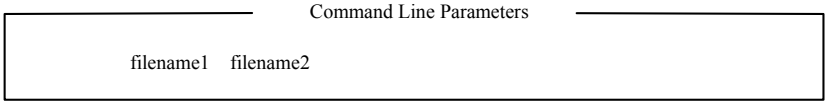

图 3.10　Options→Arguments 选择项命令窗口

6. Save options选择项

Save options 选择项的作用是将设置的选择项数据保存到磁盘文件中，默认文件名是配置文件 TCCONFIG.TC。

7. Retrive options选择项

Retrive options 选择项的作用是装入以前用 Save options 选择项保存的配置文件，为 TC 设置环境。

2.3.2　常用操作快捷键

表 3.2 列出了 TC 集成环境中常用到的操作快捷键，记住和使用这些快捷键，不需要选择菜单就能直接进行操作，加快了处理速度。

表 3.2　TC 集成环境中常用的操作快捷键及其作用

操作快捷键	作　　用
F1	打开帮助文件
F2	File→Save 项，当前编辑的文件存盘
F3	File→Load 项，打开文件
F4	Run→Go to cursor 项，使程序从光带罩住处执行到光标所在行
F5	Options→Environment/Zoom window 项，放大或复原激活的窗口
F6	交替激活窗口

（续表）

操作快捷键	作　用
F7	Run→Trace into 项，单步执行程序，跟踪函数调用
F8	Run→Step over 项，单步执行程序，不跟踪函数调用
F9	Compile→Make 项，编译并连接
F10	返回主菜单
Shift+F10	显示版本信息
Alt+F3	File→Pick 项，列出最近用户编辑的源文件
Alt+F5	Run→User screen 项，显示用户屏
Alt+F7	光标指向前一个出错处
Alt+F8	光标指向下一个出错处
Alt+F9	不作过时检查的编译，生成.obj 文件
Ctrl+F1	显示光标所在处的任意一个关键词或函数的使用信息
Ctrl+F2	Run→Program reset 项，终止调试操作
Ctrl+F3	Dubug→Call stack 项，显示程序当前运行时的函数调用序列
Ctrl+F4	Debug→Evaluate 项，检查和改变表达式的值
Ctrl+F7	Break→Watch→Add watch 项，在 watch 窗口输入要观察的表达式
Ctrl+F8	Break→Watch→Toggle breakpoint 项，设置或清除断点
Ctrl+F9	Run→Run 项，编译、连接并运行程序
Alt+C	打开 Compile 菜单
Alt+D	打开 Debug 菜单
Alt+E	进入 Edit 窗口
Alt+F	打开 File 菜单
Alt+O	打开 Options 菜单
Alt+P	打开 Project 菜单
Alt+R	打开 Run 菜单
Alt+X	退出 TC，返回 DOS
Esc	返回上一级菜单

第 3 章　Visual C++系统平台上的C语言实践

Visual C++（以下简称 VC++）为用户开发 C 程序提供了一个集成环境，这个集成环境包括：源程序的输入和编辑，源程序的编译和连接，程序运行时的调试和跟踪，项目的自动管理，而且为程序的开发提供了有关工具，并具有窗口管理和联机帮助等功能。

现在常用的是 VC++ 6.0 版本。市场上也有汉化版，但有的只是对菜单进行汉化，并非真正的全中文版，而且汉化的用语不很准确，因此许多人都使用英文版。本书以 Visual C++ 6.0 英文版为背景来介绍 VC++的上机操作。一般 VC++的不同版本的上机操作方法大同小异，掌握了其中的一种，就会举一反三，顺利地使用其他版本。

3.1　VC++的安装和启动

1. VC++的安装

VC++是 Visual Studio 中的一个组件，安装时要有 Visual Studio 的光盘，执行其中的 setup.exe 程序，并按安装向导在屏幕上的提示进行安装。

安装结束后，在 Windows 的“开始”菜单的“程序”子菜单中就会看到 Microsoft Visual Studio 子菜单。如果安装时同意建立快捷方式，则会在桌面上建立 VC++的图标。

2. VC++的启动

若桌面上建立了 VC++的图标，则可通过鼠标双击该图标启动 VC++；若没有建立相应的图标，则可以通过菜单方式启动 VC++，即从桌面上顺序选择“开始”→“程序”→Microsoft Visual Studio→Visual C++ 6.0，此时屏幕上短暂显示 VC++ 6.0 的版权页后，会弹出如图 3.11 所示的 VC++集成环境。

3. VC++的主窗口

在 VC++主窗口的顶部有标题栏；其次是 VC++的主菜单栏，其中包括“File（文件）”、“Edit（编辑）”、“View（查看）”、“Insert（插入）”、“Project（项目）”、“Build（构建）”、“Tools（工具）”、“Windows（窗口）”、“Help（帮助）”9 个菜单项；再次为工具栏，其中包括常用的工具按钮；最下方为状态栏，还有一些子窗口。

主窗口的左侧是项目工作区窗口，右侧是程序编辑窗口，工作区窗口用来显示所设定的工作区的信息，程序编辑窗口用来输入和编辑源程序。

使用 VC++集成环境上机调试程序可分成如下几个步骤：① 启动 VC++；② 生成项目；③ 生成和编辑源程序，把一个或多个源程序送到各自的文件中；④ 将源程序文件加入到项目中；⑤ 根据需要改变项目的设置；⑥ 最后编辑、连接和运行程序。下面分别进行介绍各个步骤的上机操作。

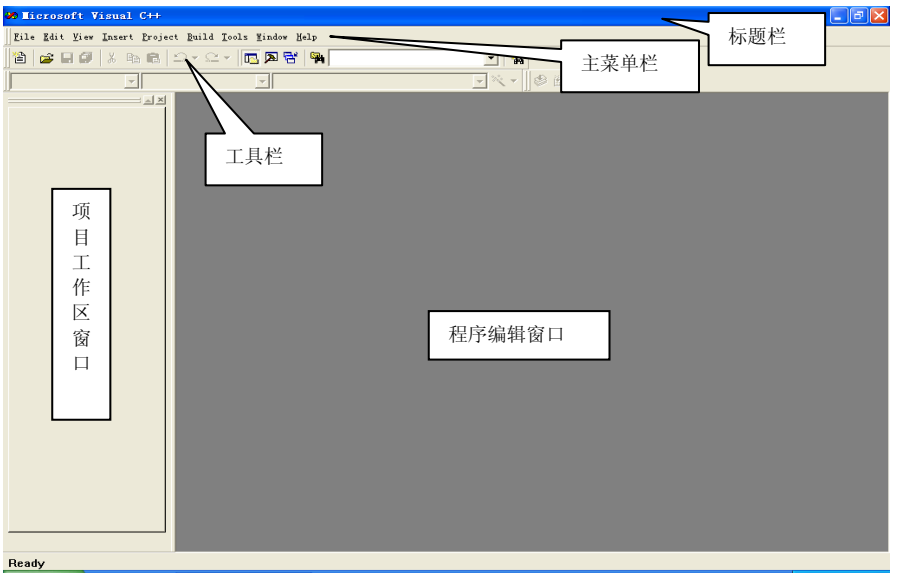

图 3.11　VC++集成环境

3.2　单个程序的操作

3.2.1　C源程序的输入和编辑

在操作之前，先确定或者建立一个存放用户 C 程序的文件夹，本书假定为 C:\VC++。

1. 新建一个C源程序

新建一个 C 源程序的步骤是：

(1) 在 VC++主窗口的主菜单栏中选择“File”菜单中的“New”命令，则出现“New”对话框。如图3.12所示。

(2) 选择对话框中的“Files”标签。

(3) 在“Files”标签左边的文件类型清单中，选择“C++ Source File”项，表示要建立新的 C++源程序文件。

(4) 在“File”标签右边的“File”文本框中输入C源程序文件名。由于 VC++6.0 既可以处理C++源程序（文件后缀名为.cpp），也可以处理 C 源程序，因此，应该给源文件名加上后缀名.c，否则系统会自动给文件加上.cpp 的后缀。例如，输入 exec1_1.c 作为新建源程序的文件名。

(5) 在“File”标签右边的“Location”文本框中输入准备编辑的源程序文件的存储路径（设为 C:\VC++）。这样，即将进行输入和编辑的源程序就以 exec1_1.c 为文件名存入在 C 盘的 VC++目录下。当然，也可以指定其他的路径名和文件名。

(6) 单击 OK 按钮后系统回到 VC++主窗口，并在窗口的标题栏中显示出用户确定的文件名和路径：C:\VC++\ exec1_1.c。可以看到光标在程序窗口闪烁，表示程序编辑窗口已激活，可以输入和编辑源程序。

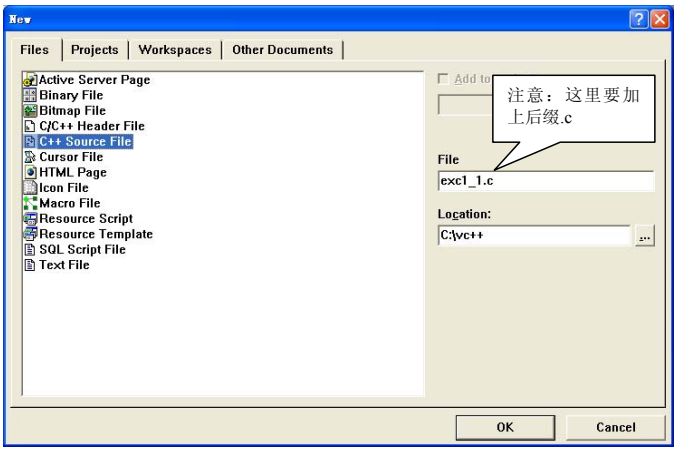

图 3.12 新建源文件对话框

(7) 输入程序。本例输入本书 2.1.5 中的最初未得到修改的程序 EXAM1_3.C，如图3.13所示。也可以单击程序编辑窗口标题栏上的“最大化”按钮，使编辑窗口最大化。

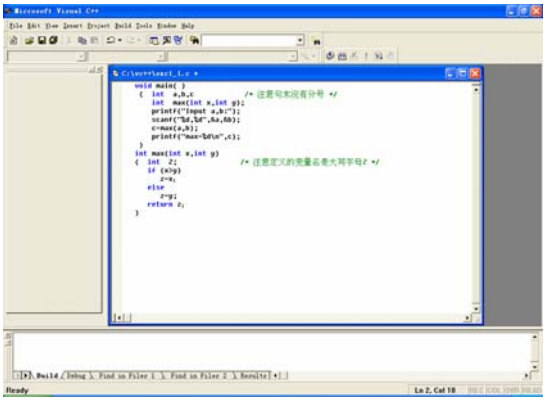

图 3.13 C 源程序输入界面

在输入过程中如果出现某些错误，可以利用读者十分熟悉的具有 Windows 风格的全屏幕编辑方法进行修改编辑，包括输入中文注释，这比 TC 要方便得多。在图 3.13 最下部的状态栏，用“Ln 2, Col 18”显示了光标当前的位置在第 2 行第 18 列。当光标位置改变时，显示的数字也随之改变。在对程序进行定位编辑时，这个显示是有用的。

小窍门：如果后面要输入的源程序文件格式同当前源程序文件格式基本一样，不妨使用 Save As（另存为）项，将当前文件内容转存为要输入的文件，再进行修改，可省去很多输入的繁杂工作。方法是通过 File（文件）→Save As（另存为）将它以另一个文件名另存（如以 excl_2.c 名字另存），这样就生成了一个新文件 excl_2.c。

(8) 保存程序。输入程序后，如果经检查无误，接下来将源程序保存在前面指定的文件中。在主菜单栏中选择 File，并在其下拉菜单中选择 Save（保存）项。如图 3.14 所示。

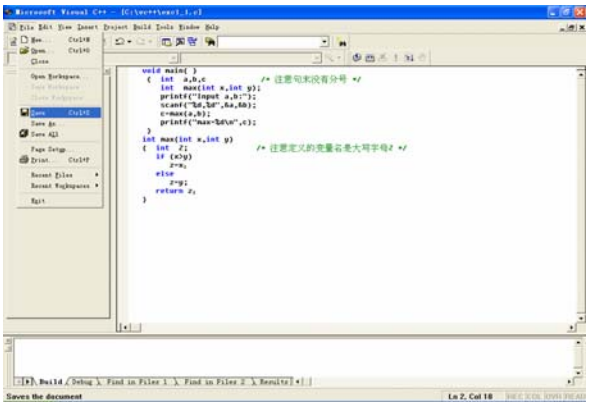

图 3.14　保存当前编辑的 C 源程序文件

也可以用 Ctrl+S 快捷键来保存文件。如果不想将源程序存放到原先指定的文件中，可以不选择 Save 项，而选择 Save As（另存为）项，并在弹出的 Save As（另存为）对话框中指定文件路径和文件名。

2. 已有C源程序的打开

如果希望打开已保存在磁盘上的源程序文件，并对它进行处理，有两种操作方法：

(1) 在“Windows 资源管理器”或“我的电脑”中按路径找到已有的 C 程序名（如 excl_1.c）。双击此文件名，则自动进入 VC++集成环境，并打开该文件，程序显示在编辑窗口中。

(2) 单击工具栏中的 Open 小图标或按 Ctrl+O 键，也可以选择 File（文件）→Open（打开），都可以打开 Open 对话框，从中选择所需的文件。

小窍门：如果要打开的源程序文件是不久前打开过的，通过 File（文件）→Recent File（最近的文件），会出现最近用过的源文件列表，从中选择所要打开的文件，此方法比较快捷。

3.2.2　单程序的编译、连接和运行

1. 单程序的编译

在编译和保存了源文件（如 excl_1.c）以后，若需要对该源文件进行编译，则其操作步骤是：

(1) 单击主菜单栏中的 Build（编译），在其下拉菜单中选择 Compile excl_1.c（编译

excl_1.c)，如图3.15所示。由于建立或保存文件时已指定了源文件的名字 excl_1.c，因此在 Build 菜单的 Compile 项中就自动显示了当前要编译的源文件名 excl_1.c。

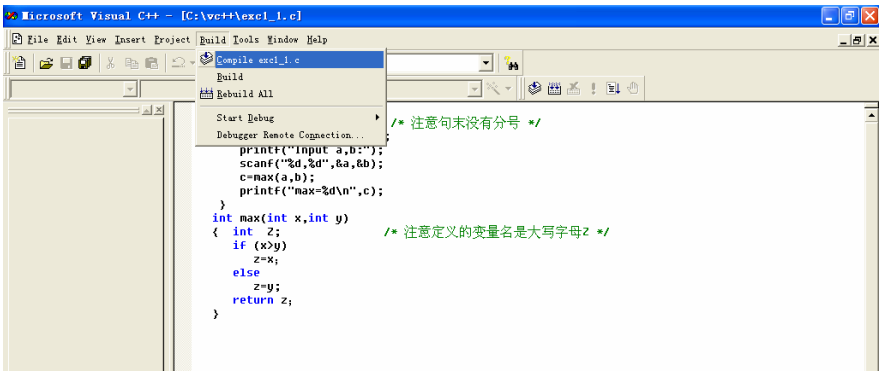

图 3.15　选择编译 C 程序文件

(2) 在单击编译命令后，屏幕上出现一个对话框，内容是 This build command requires an active project workspace, Would you like to creat a default project workspace ?（此编译命令要求一个有效的项目工作区，你是否同意建立一个默认的项目工作区？），如图3.16所示。单击是（Y）按钮，表示同意由系统建立默认的项目工作区，然后开始编译。

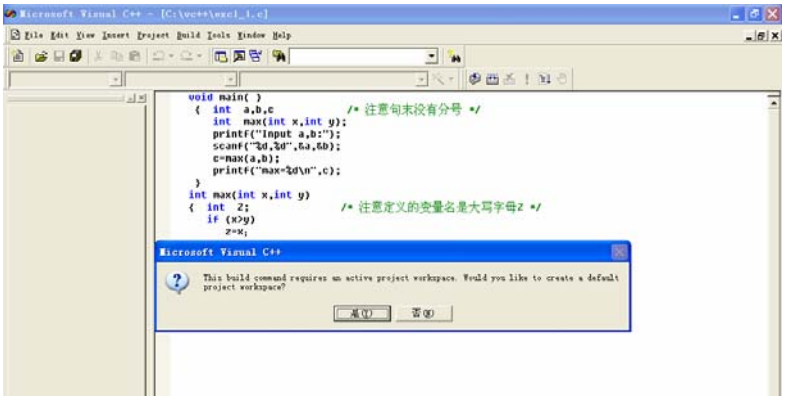

图 3.16　编译时要求用户同意默认的项目工作区

也可以不用选择菜单的方法，而用 Ctrl +F7 快捷键来完成编译。

(3) 在进行编译时，编译系统检查源程序中有无语法错误，然后在主窗口下部的调试信息窗口输出编译的信息。如果有错，就会指出错误的位置和性质，如图3.17所示。

图3.17下部的调试信息窗口显示文件 exc1_1.obj 有“8 error(s), 2 warning(s)”（8 个致命错误，2 个警告错误），将导致“Error executing”（执行出错）。

(4) 分析程序中的错误原因，找到解决的办法。

编译系统能检查出程序中的语法错误，语法错误分为两类：一类是致命错误，以“error”表示，如果程序中有这类错误，就通不过编译，无法形成目标程序，更谈不上运行；另一类是轻微错误，以“warning”（警告）表示，这类错误不影响生成目标程序和可执行程序，但有可能影响运行的结果，因此也应当改正，使程序既无“error”，又无“warning”。

程序 exc1_1.c 的错误包括：

① 程序的第 3 行类型定义前缺乏“；”号（missing ';' before 'type'），这正是 2.1.5 节中所预料的；

② 程序第 4、5 行中“printf”和“scanf”未定义，这是因未将头文件 stdio.h 包含进来所致；

③ 程序第 12、15 行由于变量 z 未定义，以致出现一连串错误；

④ 程序第 13、16 行还有一个丢失“;”号的错误。仔细观察会发现第 13、16 行末尾用的是中文的“；”。

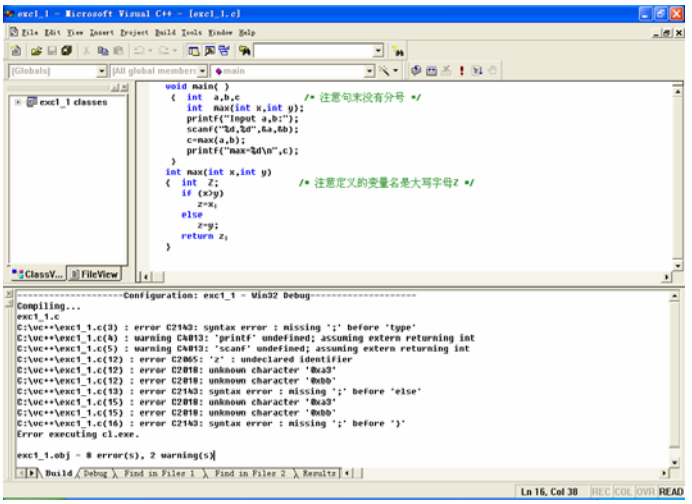

图 3.17　编译时出现相关信息的界面

可见，在分析编译报错信息时，应检查出错点的上下行。

(5) 根据系统的提示，进行改错。双击调试信息窗口中的第 1 个报错行，这时在程序窗口中出现一个粗箭头指向被报错的程序行（第 3 行），以提示改错位置，如图3.18所示。

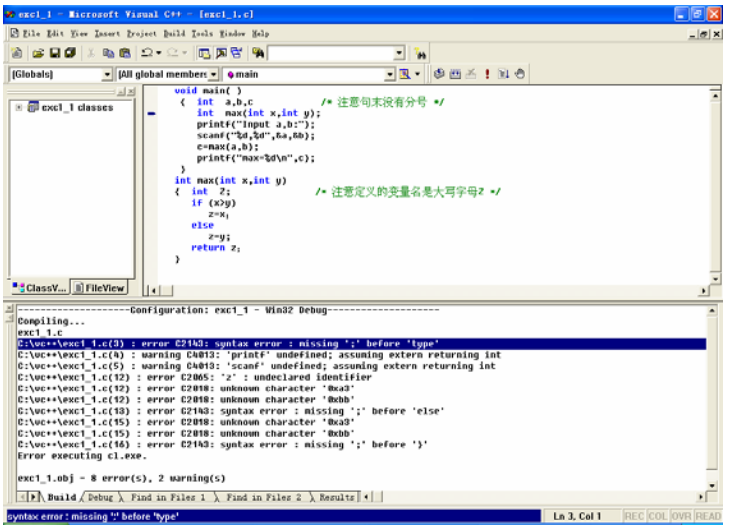

图 3.18 程序改错提示修改位置

要去掉该错误，应在程序第 2 行末尾添加一个“;”；

类似地，还要修改 4 个地方：

① 程序开头插入预处理命令：#include "stdio.h"；

② 在程序第 10 行中用小写 z 替换 Z；

③ 在程序第 13 行末尾用“;”替换“；”；

④ 在程序第 16 行末尾为“；”替换“;”。

(6) 重新编译。修改后重复步骤(1)和(2)，屏幕显示如图 3.19。

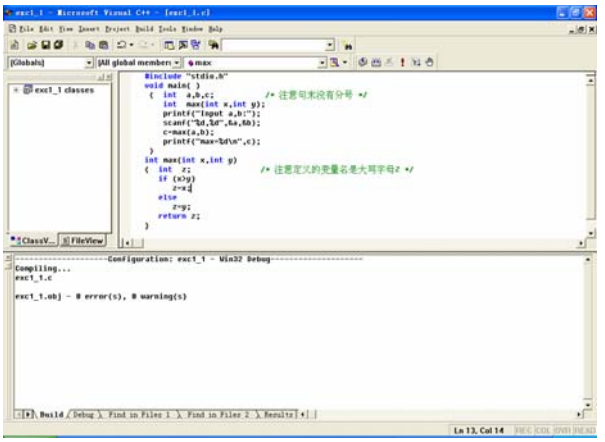

图 3.19 修改程序后重新编译时出现相关信息的界面

系统显示“0 error(s), 0 warning(s)”，说明既没有致命错误，也没有警告错误，编译成功，这时产生了一个 exc1_1.obj 文件。

至此完成了 C 程序的编译操作。可见，如果在编写和输入程序时就严格按照语法的要求去做，有可能一次编译成功，而省去调试改错的环节。所以，程序设计看起来只是上机操作，实际上很多工夫是在上机之外要做的，对编写规模较大的程序更是如此。

2. 单程序的连接

通过编译得到目标程序后，接下来是对程序进行连接。选择 Build→Build exc1_1.exe。由于已生成了目标程序 exc1_1.obj，编译系统据此确定在连接后应生成一个名为 exc1_1.exe 的可执行文件，因此菜单中显示了此文件名，如图 3.20 所示。也可以直接按 F7 实现连接功能。

完成连接后，在调试信息窗口中显示连接时的信息，报告“exc1_1.exe - 0 error(s), 0 warning(s)”，说明没有发现错误，生成了一个可执行文件 exc1_1.exe。

程序的编译与连接可分别进行，也可以选择菜单 Build→Build（或按 F7 键）一次完成。对于初学者来说，分步进行程序的编译与连接，有利于程序改错。等到编程熟练，程序难得出现语法错误时，可以一步完成编译与连接。

3. 单程序的执行

有可执行文件 exc1_1.exe 之后，就可以直接执行。选择 Build→!Execute exc1_1.exe（执行 exc1_1.exe）或者单击工具栏中的[!]按钮或者按组合键 Ctrl+F5 都能实现程序的执行，如图 3.21 所示。

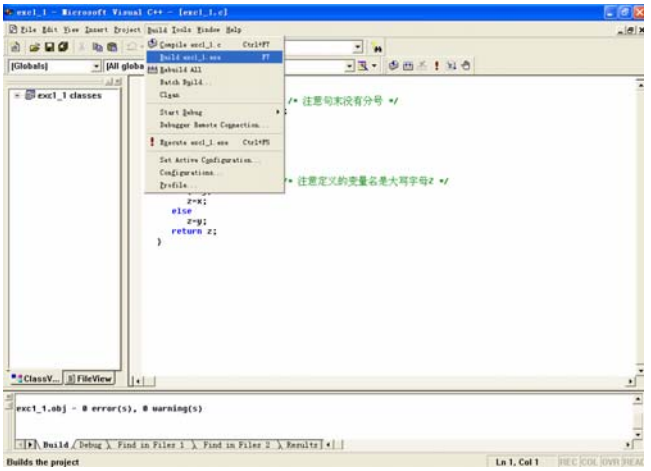

图 3.20　编译通过后进行连接的操作

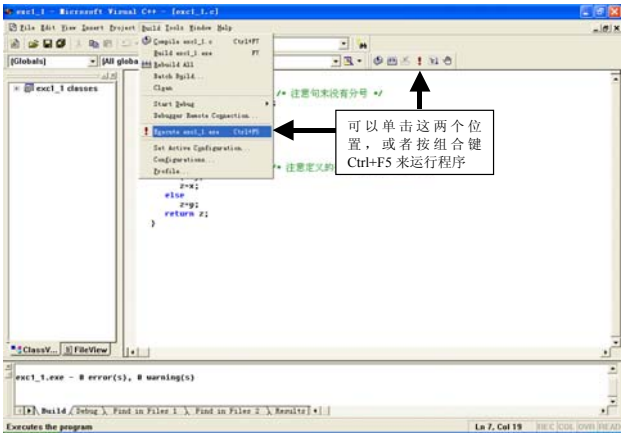

图 3.21　连接通过后进行程序运行的操作

此时，屏幕切换到输出结果的窗口。按照程序，系统先显示要求输入数据的信息：“Input a,b: ”，出现光标闪烁，要求用户输入。现输入 3, 7 按回车键，即马上显示运算结果，如图 3.22 所示。在输出窗口的最后一行，显示“Press any key to continue”（按任意键以便继续），这是系统自动加上的一行信息。此时随便按一个键，则输出窗口消失，回到 VC++的主窗口，用户可以继续对源程序进行修改补充或进行其他的工作。

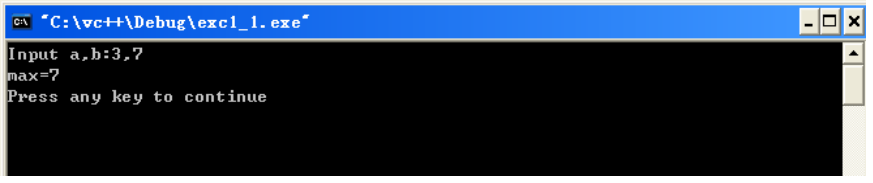

图 3.22　程序执行时的输出窗口

如果已完成对一个程序的操作，且不再对它进行其他的处理，应当选择 File（文件）→ Close Workspace（关闭工作区），以结束对该程序的操作。

3.3 包含多个文件的程序上机操作

如果一个程序包含多个源程序文件，则需要建立一个项目文件（.prj 文件）。一个项目文件中可以包含多个文件（包括.c 源文件和.h 头文件）。项目文件只能放在项目工作区中，因此还要建立项目工作区。在编译时，系统会分别对项目文件中的每个文件进行编译，然后将所得到的目标文件连接成为一个整体，再与系统的有关资源连接，生成一个可执行文件。

实际上，VC++的所有程序文件都要放在相应的工作区的项目文件中。在前面介绍的单程序的操作中，省略了用户建立项目工作区和项目文件的操作，而由系统自动建立。

有两种实际操作方法：一种是由用户建立项目工作区和项目文件；另一种是用户只建立项目文件，而不建立项目工作区，由系统自动建立项目工作区。

3.3.1 由用户建立项目工作区和项目文件

具体操作步骤是：

(1) 分别编辑好同一个大程序中的各个源程序文件，并存放在指定的目录下。

仍以 2.1.6 节的三个源文件 exam1.c、exam2.c 和 exam3.c 作为例子，假定已把它们保存在 C:\VC++文件夹下。

(2) 建立一个项目工作区。

在 VC++主窗口中选择 Files（文件）→New（新建），在弹出的 New（新建）对话框中单击上部的选项卡 Workspace（工作区），表示要建立一个新的项目工作区，如图 3.23 所示。

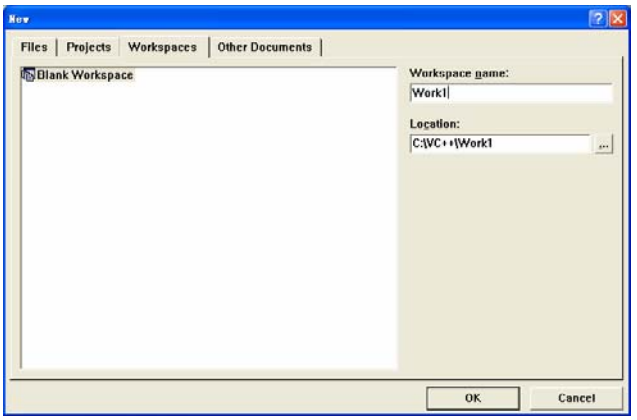

图 3.23 新建项目工作区对话框

在对话框右部 Location（位置）文本框中输入指定的文件目录（如 C:\VC++），在 Workspace name（工作区名字）文本框中输入用户指定的工作区的名字（如 Work1），此时，Location（位置）文本框中会自动添加项目工作区名，如图 3.23 所示。当然，也可以指定为其他工作区名字或其他目录。

单击右下部的 OK 按钮，返回 VC++主窗口。

(3) 建立项目文件。

选择 Files（文件）→New（新建），在弹出的 New（新建）对话框中单击上部的选项卡 Projects（项目）。表示要建立一个项目文件，如图 3.24 所示。

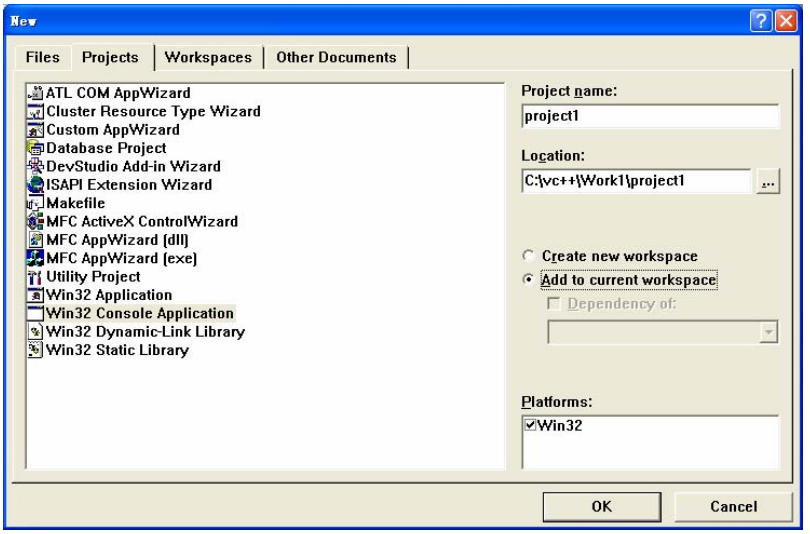

图 3.24　新建项目文件对话框

在对话框左边的列表中选择 Win32 Console Application 项，并在右部的 Location 文本框中输入项目文件的位置，即文件路径，现在输入 C:\VC++。也可以单击…按钮，在出现的目录选择框中选定项目文件所在的工作区，如图3.25所示，本例选定 Work1 工作区目录。

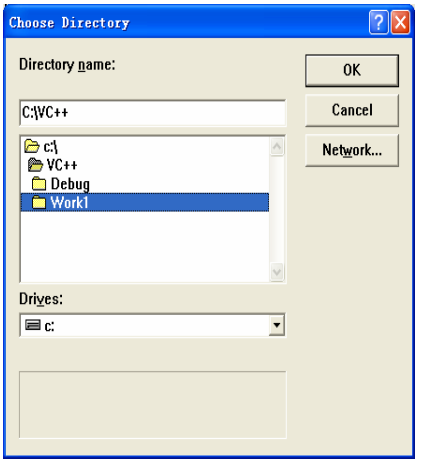

图 3.25　在目录对话框中选定 Work1 工作区

在 Project name 文本框中输入指定的项目文件名，现在输入 project1，此时，Location 栏中的内容自动变为 C:\VC++\Work1\project1，表示已确认项目文件 project1 存放在工作区 Work1 中。选中窗口右边的单选项 Add to current workspace（添加到现有工作区），表示新建的项目文件是放到刚才建立的当前工作区 Work1 中的。然后单击 OK 按钮，此时弹出一个如图3.26所示的对话框。在其中选中 An empty project 单选钮，表示新建立的是一个空的项目，单击 Finish（完成）按钮，系统弹出一个 New Project Information（新建工程信息）对话框（如图3.27 所示），显示了刚才建立的项目的有关信息。

在其下方可以看到项目文件的位置（文件路径为 C:\VC++\Work1\project1），确认后单击 OK 按钮，此时又回到 VC++主窗口，可以看到左边窗口中有一个 Workspace 窗口，单击其中的 File View 选项卡，窗口内显示“Workspace ‘project1’: 1 project[s]”，表示工作区 Work1 中有一个项目文件，其下一行为 project1 files，表示项目文件 project1 中的文件现在为空，如图3.28所示。

(4) 在项目文件中添加源程序文件。方法是在 VC++主窗口中选择 Project（工程）→Add To Project（添加到项目中）→Files，如图3.29所示。

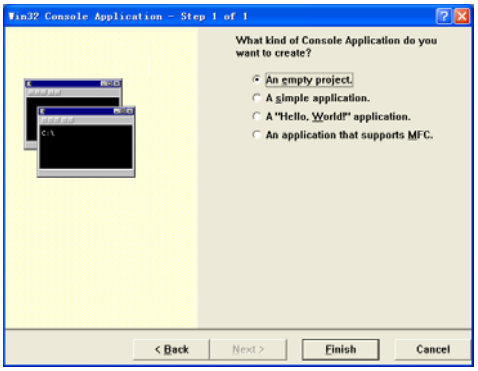

图 3.26 选择新建项目文件内容的对话框

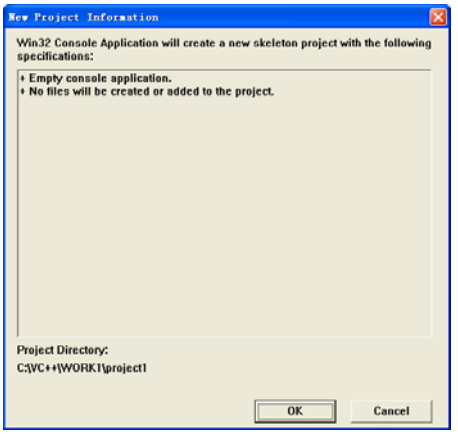

图 3.27 新建项目文件建立后显示项目的有关信息

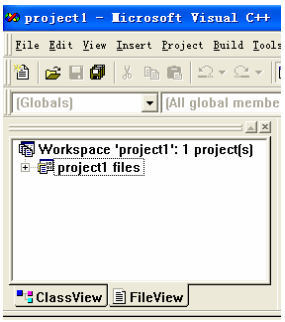

图 3.28 返回后项目工作区窗口的显示

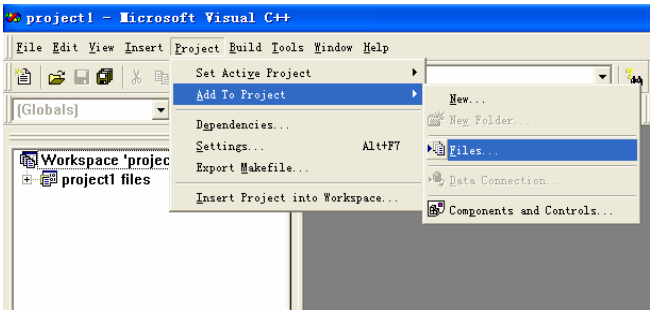

图 3.29 在项目文件中添加源程序文件

选择 Files 命令后，屏幕上出现 Insert Files into Project 对话框。在其上部的路径栏中找到源文件 exam1. c、exam2. c 和 exam3. c 所在的子目录，并选中这 3 个文件，如图3.30所示。

单击 OK（确定）按钮，把这 3 个文件添加到项目文件 project1 中。此时，回到 VC++主窗口，再观察 Workspace 窗口，单击其下部的 File View 选项卡，按 project1 files→Source Files 的顺序打开 Source Files 文件夹，可看到项目文件 project1 中包含了源程序 exam1. c、exam2. c 和 exam3. c，如图3.31所示。

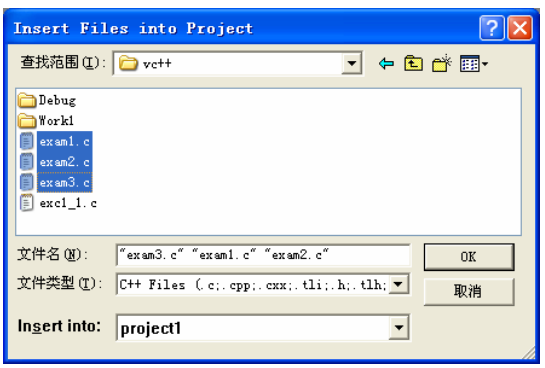

图 3.30 选择要添加的源程序文件

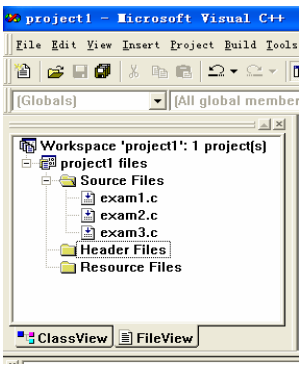

图 3.31 项目工作区窗口显示所包含的源程序文件

(5) 编译和连接项目文件。

对项目文件的编译和连接，就是对项目中的源程序文件（本例是 exam1. c、exam2. c 和 exam3. c）进行统一的编译和连接。具体步骤是：

① 在 VC++主窗口中选择 Build（编译）→ Build project1.exe（构件 project1.exe），如图3.32所示。

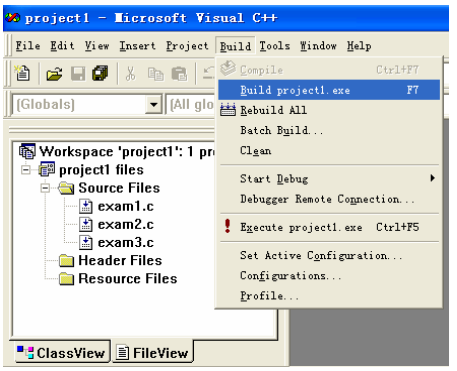

图 3.32　编译项目文件的菜单操作

单击 Build project1.exe 项后，系统对整个项目文件进行编译和连接，在窗口的下部会显示编译和连接的信息。如果程序有错，会显示出错信息；如果程序无错，会生成可执行文件 project1. exe。本例有意识地在三个程序中设置了一些错误，项目文件编译不能通过，系统显示如图3.33所示。

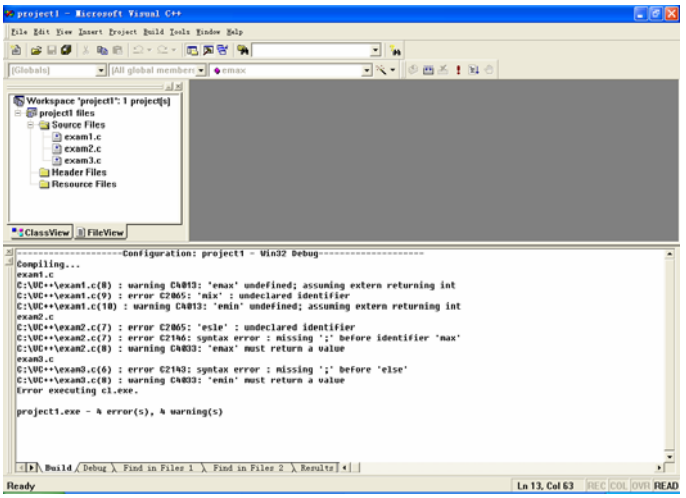

图 3.33　编译项目文件时出现的出错信息

② 按照出错提示分别分析和修改各个源程序中的错误。方法是双击调试信息窗口中的某一出错提示，系统立即将相应的源程序在程序编辑窗口中列出，并用箭头指示出错的程序行，如图3.34所示。该出错信息为警告信息，指出“'emax' undefined”，经分析知道，exam1. c 中要调用其他文件中的 emax( )函数，但没有对该函数进行声明。为此，在该程序的第 2 行插入一个程序行：

```
extern int emax(int, int);
```

类似地，逐个将各个程序中的错误给以改正。如果希望审视各个程序清单，则可双击项目工作组窗口中显示的源程序名，程序编辑窗口即显示相应的程序清单，以便用户检查和改错。

③ 改错后，再按步骤①和②操作，直到消除全部错误，此时调试信息窗口将显示：

```
project1.exe - 0 error(s), 0 warning(s)
```

说明已生成可执行文件 project1.exe。

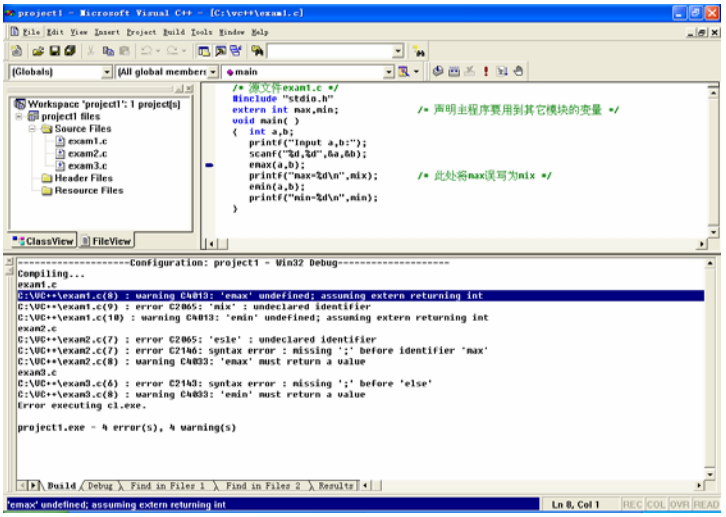

图 3.34 针对出错信息对源程序进行调试和修改

(6) 执行可执行文件。选择 Build（编译）→Execute project1.exe（执行 project1.exe），就可执行 project1.exe。在运行时输入所需的数据，如图3.35所示。

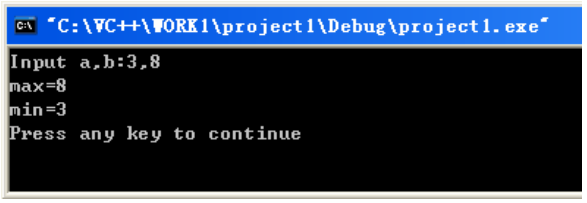

图 3.35 执行可执行文件 project1.exe 时的输出窗口

也可以单击工具栏中的“!”按钮（BuildExecute）或者按快捷键“Ctrl+F5”，直接编译与运行源程序。

3.3.2 用户只建立项目文件

上面介绍的方法是先建立项目文件工作区，再建立项目文件，步骤比较多。可以采取简化的方法，即用户只建立项目文件，而不建立项目工作区，由系统自动建立项目工作区。

在本方法中，保留 3.3.1 节中介绍的第(1)、(4)、(5)、(6)步，取消第(2)步，修改第(3)步，具体步骤如下：

(1) 分别编辑好同一个程序中的各个源程序文件，同 3.3.1 节中的第(1)步。

(2) 建立一个项目文件（不必先建立项目工作区）。

在 VC++主窗口中选择 Files（文件）→New（新建），在弹出的 New（新建）对话框中单击上部的选项卡 Projects（工程），表示要建立一个新的项目文件，如图3.36所示。

在对话框左边的列表中选择 Win32 Cunsole Application 项，并在右部的 Project name（工程名）文本框中输入指定的项目文件名（Project2），在右部中间的单选钮处默认选定了 Create new workspace（创建新工作区），这是由于用户未指定工作区，系统会自动开辟新工作区。

单击 OK（确定）按钮，出现如图2.26所示的 Win32 Cunsole Application-step 1 of 1 对

话框，选择右部的单选钮 An empty project，单击 Finish（完成）按钮后出现 New Project Information（新建成工程信息）消息框，如图3.37所示。

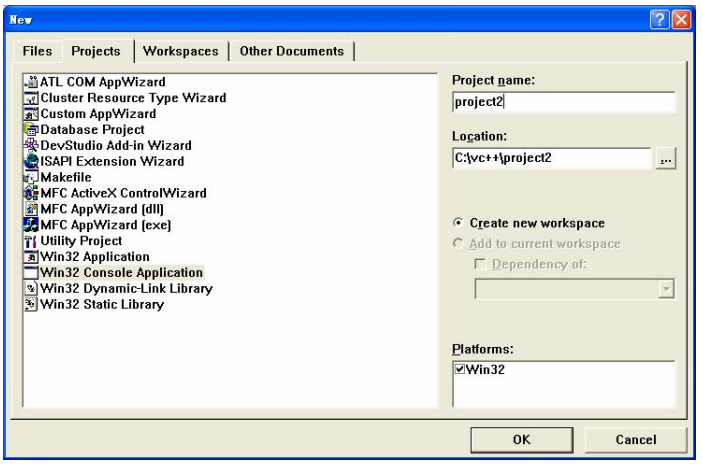

图 3.36　只建立项目文件 New 对话框

从它的下部可以看到项目文件的路径（即“工程目录”）为 C:\VC++\project2。单击 OK（确定）按钮，在弹出的 VC++主窗口中的 Workspace 窗口的下方单击 File View 按钮，窗口中显示“Workspace ‘project2’: 1 project[s]”，如图3.38所示，说明系统已自动建立了一个工作区，由于用户未指定工作区，系统就将项目文件名 project2 同时作为工作区名。

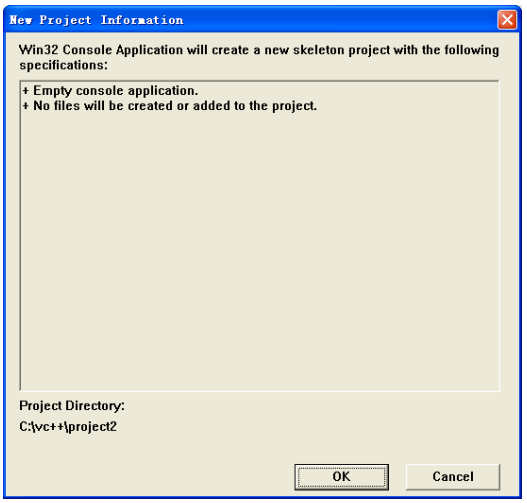

图 3.37　新建项目文件 project2 建立后显示项目的有关信息

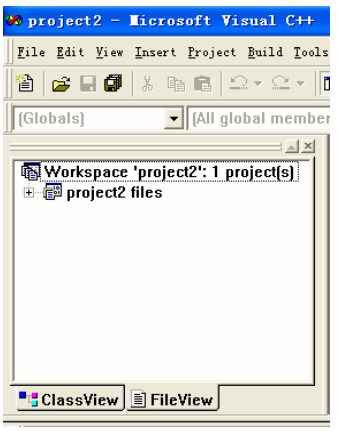

图 3.38　返回后项目工作区窗口的显示

(2) 向此项目文件添加源程序文件，步骤与 3.3.1 节方法中的第(4)步相同。

(3) 编译、调试和连接项目文件，步骤与 3.3.1 节方法中的第(5)步相同。对项目文件 project2 进行编译连接后生成 project2.exe，如图3.39所示。

(4) 执行可执行文件，步骤与 3.3.1 节方法中的第(6)步相同。

显然，这种方法比前面的方法简单一些。

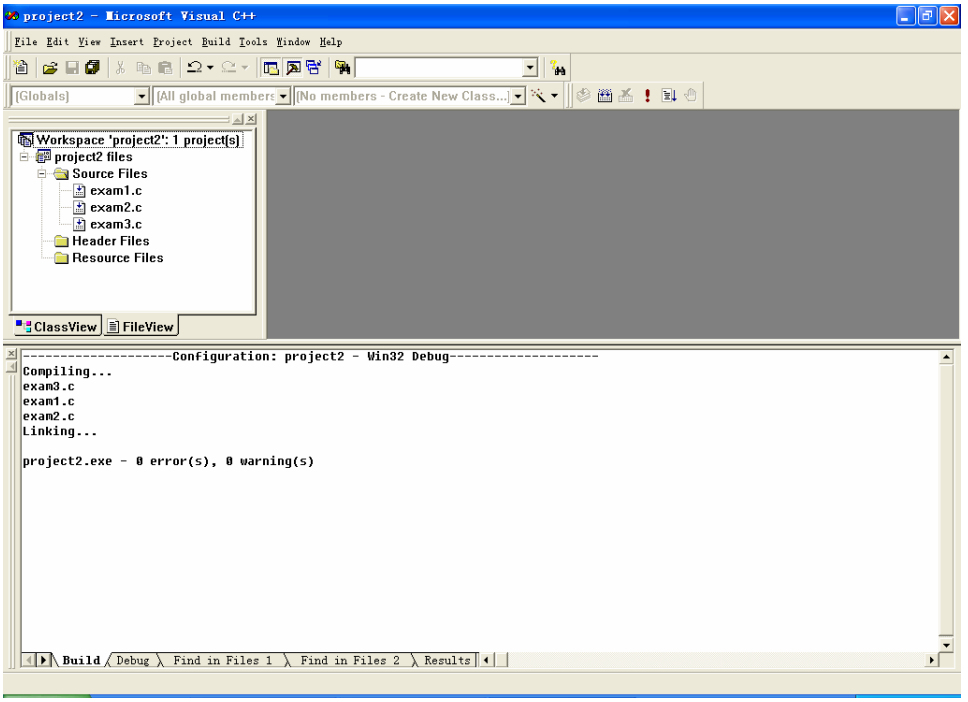

图 3.39　对项目文件 project2 进行编译连接后生成 project2.exe

在介绍单文件程序时，为了尽量简化手续，没有建立工作区，也没有建立项目文件，而是直接建立源文件。实际上，在编译每一个程序时都需要一个工作区，如果用户未确定，系统会自动建议工作区，并赋予它一个默认名（此时以文件名作为工作区名）。

3.3.3　打开已存在的项目文件

可用两种方法打开已存在的项目文件。

(1) 在 VC++主窗口中选择“File”菜单中的“Open workspace”命令，如图 3.40 所示。然后在弹出的对话框中选择要打开的工作区，如 Work1 工作区，如图 3.41 所示。再在打开的工作区中选择要打开的项目文件夹和所选定的项目文件，如图 3.42 和图 3.43 所示。

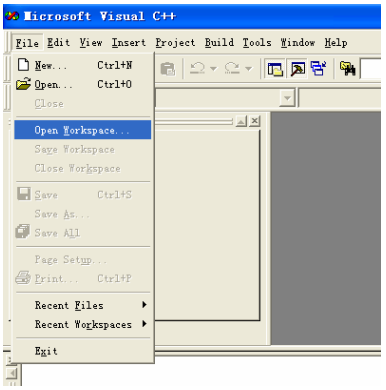

图 3.40　打开已创建的项目文件

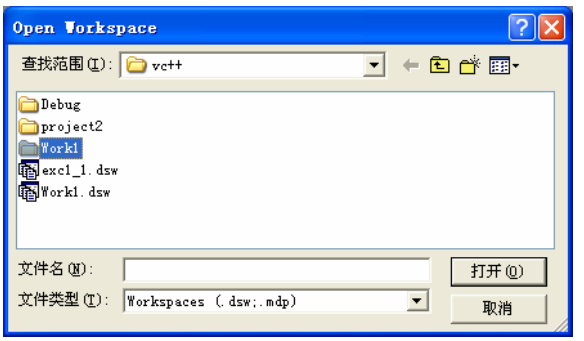

图 3.41　在 Open Workspace 对话框中选择要打开的工作区 Work1

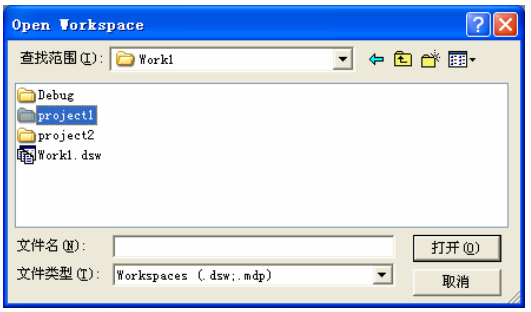

图 3.42　选择要打开的项目文件 project1

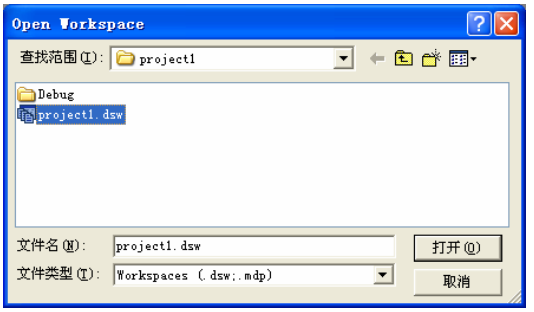

图 3.43　打开选定的项目文件 project1, dsw

(2) 在 VC++主窗口中选择“File”菜单中的“Recent workspaces”命令，然后再选择相应的项目文件，如图3.44所示。

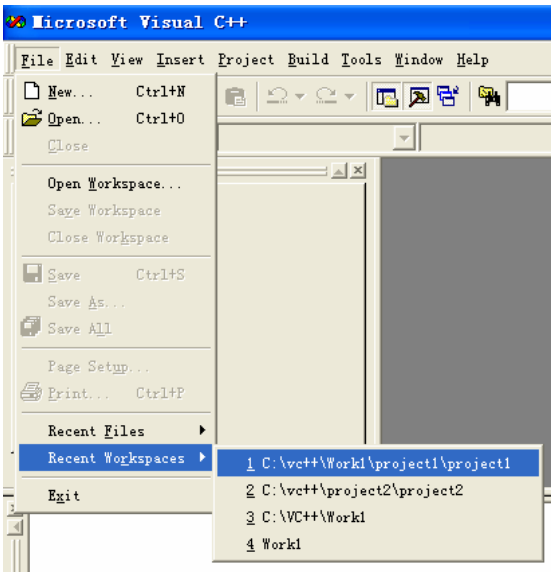

图 3.44　选择 Recent workspaces 打开项目文件 project1

注意：在调试一个应用程序时，VC++集成环境一次只能打开一个项目文件。当一个程序调试完成后，要开始输入另一个程序时，必须先关闭当前的项目文件，然后为新源程序建立一个新的项目文件。关闭当前的项目文件的方法是，选择“File”菜单中的“Close workspace”命令。

3.3.4　退出VC++集成环境

在 VC++主窗口中选择“File”菜单中的“Exit”命令，即可以退出 VC++集成环境。

3.4　VC++环境下C程序的动态调试

如果程序执行结果出错，或者对程序执行结果产生怀疑，在 VC++环境下，可采用与 Turbo C 一样的操作方法，对程序执行过程进行监视和调试。VC++同样提供了单步调试和断点调试的方法。下面分别给以介绍。

3.4.1 单步调试法

下面仍然结合 2.2.1 节中 1 个简单的实例，即求两个数的乘积的程序，说明如何在 VC++ 环境下使用单步跟踪法调试程序。重写该程序如下：

```
main()
{   int x,y,z;
    printf("Please input two number:");
    scanf("%d,%d",&x,&y);
    z=x*y;
    printf("z=%d\n",z);
}
```

程序很简单，上机很顺利地通过编译连接，第 1 次运行情况如下：

```
Please input two number: 10，20
z=200
```

下面采用单步运行。选择菜单 Build→Start Debug，在展开的下级菜单中选“Step Into”（如图 3.45 所示）。

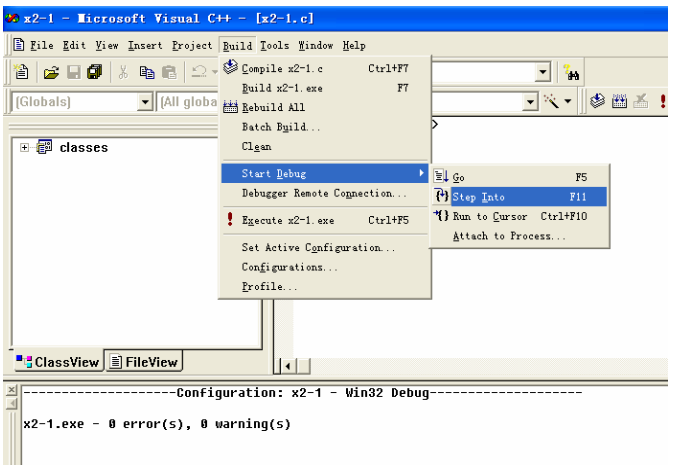

图 3.45 单步执行的起始操作

此时，程序开始单步执行，屏幕上会出现命令提示符窗口，并且程序编辑窗口的左端会有一个箭头指向下一条要执行的语句“{ int x, y, z;”，（如图 3.46 所示）。随后，应该选择菜单 Build 下的“Step Over (F10)”项或按 F10 键，可以进行程序单步运行。不断选择“F10”，程序会一步一步地被执行。注意，之所以不再选择“Step Into”项，是因为该选项在遇到函数调用时会进入函数内部，并从函数头开始单步执行。如果遇到的是库函数如 printf()、scanf() 等调用，程序会跟踪至库函数内部，而使调试不能顺利进行。因此，除非是对自定义函数的调试，一般不要使用这一操作。

当执行到“scanf("%d,%d",&x,&y);”时，用户应在命令提示符窗口上输入数据。此时，在屏幕下方的 Variables 窗口中的 Auto 文件夹会显示&x、&y 的数值，以及执行 printf()后的返回值，输入数据回车后，Variables 窗口中会报告此时变量 x、y 的数值，以及执行 scanf()后的返回值。如果希望监视 x*y 的数值，可在 Watch1 窗口中键入要观察的表达式 x*y，如图3.47

所示。其中 x*y 和 z 的值都显示正确的结果 12 000，并没有出现 Turbo C 环境中的错误，其原因是 VC++6.0 中的 int 型数据占 4 个字节，因而没有溢出。

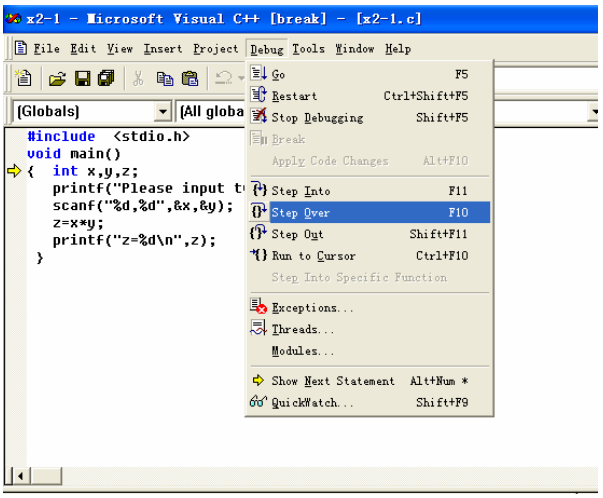

图 3.46　选“Step Over (F10)”项执行单步操作

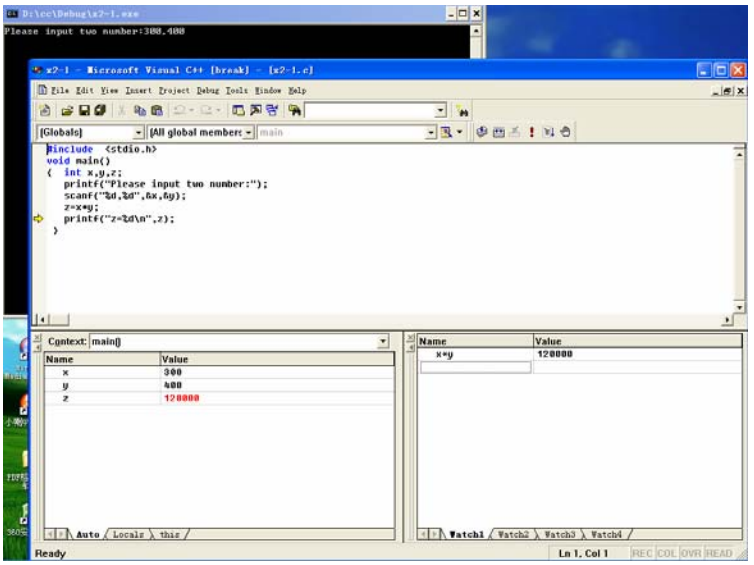

图 3.47　监视程序中有关变量和表达式的值

当执行到主函数体的末尾花括号处时，可连续按 F10 让程序回到单步开始状态，也可选择 Build→Restart 或 Build→Stop Debugging，让程序回到单步开始状态。

选择菜单 Edit→Undo 或按工具栏上的相应按钮，就能消除单步状态。

3.4.2　设置断点法

将鼠标停留在程序中希望作为断点要被暂停的那一行，选择工具栏中的手形按钮“Insert/Remove Breakpoint[F9]”，如图 3.48 所示，就可以添加一个断点。仿照此办法可设置多个断点。

选择菜单 Build→Go 或按 F5，就可以执行程序至断点处暂停。此时可观察 Variables 窗

口或 Watch 窗口中的内容。如果后面还有设置的断点，则可按 F5，程序会一直运行到下一个断点处；也可按 F10，进行单步运行。当然，若无断点，则按 F5 可执行到程序结束。

图 3.48　添加一个断点的操作

如果要消除已经被设置的断点，可将光标移至已设断点的行，选择工具栏中的手形按钮，即可删除该行断点。

小窍门：实际上还有一个更简便的断点调试方法——光标控制法。即先将光标定位于想要暂停的程序行，然后选 Build→Run to Cursor，程序会一直运行到当前光标所在处停止，并在该程序行首出现一个箭头标志。

第 4 章　常见上机错误与纠正

C 程序设计过程中经常出现的错误有编译错误、连接错误、运行错误和操作错误。下面根据作者长期的教学经验，总结分析了一些常见的错误类型，分类提供给读者参考。

4.1　常见编译错误与纠正

编译错误也称为语法错误，指违背了 C 语言语法规定的错误。对于这类错误，编译程序一般能给出“出错信息”，并且告诉在哪一行出错。可以通过阅读分析“错误信息”提示（见附录 A），对该行或上下几行进行语法检查以排除错误。下面是常见的编译错误及其纠正方法。

(1) 变量在使用前未定义。如：

```
void main ()
{    x = 3;
     printf("%d\n",x);
}
```

说明：C 语言要求对程序中用到的每一个变量都要预先定义其类型，上面程序中没有对 x 进行定义，应在函数体的开头加上 `int x;`。

又如：

```
struct student
{    long       num;
     char       name[20];
     char       sex;
     int    age;
};
student.num=810234;
strcopy( student.name, "Liuwzi" );
student.sex='M';
student.age=20;
```

说明：上面程序中虽然定义了一个结构体类型，但是并没有定义结构体变量。本例的错误是混淆了结构体类型和结构体变量的区别，不合法地对结构体类型赋值。因此，应在程序中定义结构体变量后再进行结构体成员的赋值，改为：

```
struct student
{    long       num;
     char       name[20];
     char       sex;
     int    age;
};
struct student stu;
stu.num=810234;
strcopy( stu.name, "Liuwzi" );
stu.sex='M';
stu.age=20;
```

(2) 错误地使用了空格。

例如，在对 C 程序中的任何部分作注释时，用 / *…* / 来表示。这时/与*之间都不应当有空格。在关系运算符<=，>=，==，&&，||和!=中，两个符号之间也不允许有空格。另外，在格式符如%d、%f，转义符/n、/t 之中都不允许有空格。

(3) 在应该使用分号的地方漏了分号。

① 语句没有以分号结束。如：

```
a=3      /* 正确的语句为  a = 3; */
b=4;
```

此时，C 语言将上面两个语句看做一个语句而带来语法错误。

② 如果用复合语句，则有时候往往漏写最后一个语句的分号。如：

```
{   t=a;
    a=b;
    b=t      /* 正确的语句为  b=t; */
}
```

③ 在结构体类型变量定义时，没有写第二个花括号后面的分号。如：

```
struct st
  {    char data;
       int next;
  }         /* 末尾缺分号 */
```

④ do～while 语句末尾漏掉了分号。如：

```
do{ sum+=a[i];
    i++;
  }while(i<=n)
```

⑤ 错用中文标点分号。如：

```
m='a';       /* 此句应改为 m='a'; */
```

(4) 在不该出现分号的地方加了分号。

① 函数定义时，不该加分号时加分号。如：

```
int  max(int x);     /* 函数原型定义，此处不应有分号 */
{ … }
```

② 在#include、#define 等编译预处理命令末尾加了“;”号。

③ 在流程控制语句中添加不必要的“;”号，有时不一定造成编译错误，只可能改变算法而出现运行错误；但在 if 语句中添加不必要的“;”号，会使 else 缺少与之配套的 if 语句，而出现语法错误。如：

```
if( x>y ) z=1;
else if( x==y ) z=0; ; /* 此处多了一个";"号，其后的else被排除在if( x==y )语句之外 */
else z=-1;
```

(5) 不能正确区分字符与字符串。

C 语言中规定字符型常量用单引号括起来，而字符串常量则用双引号括起来，并且字符串只能存放在字符型数组中。例如，以下语句在编译时会出错：

```
char m ;
m="a" ;          /* 此句应改为 m='a'; */
```

(6) 错误地定义或引用数组。

C 语言规定，定义或引用数组时必须要在数组名后加方括号，二维数组或多维数组的每一维数据都必须分别用方括号括起来。例如，以下写法都将造成编译时出错：

```
int x=8, a[x];  /* 数组的长度应为常量，此处应为 int a[8]; */
int a(8);       /* 应为  int a[8]; */
char b[5,2];    /* 应为  char b[5][2]; */
a(1) = 3;       /* 应为  a[1] = 3; */
```

(7) 误以为数组名可代表数组中全部元素。如：

```
int a[5]={0, 2, 4, 6, 8};
printf("%d,%d,%d,%d,%d\n",a);
```

试图用数组名代表数组中全部元素而出错。在 C 语言中，数组名代表数组首地址，不能通过数组名输出 5 个元素，可用循环来实现一维数组元素的输出。

(8) 混淆字符数组与字符指针的区别。如：

```
void main ()
{   char    str[20];
    str="C Program";
    printf("%s\n", str);
}
```

str 是数组名，代表数组首元素地址。编译时，对 str 数组分配了一段内存单元，因此程序运行期间，str 是一个常量，不能再被赋值，所以 str＝“C Program”不符合语法。若把

```
char     str[20];
```

改为

```
char     *str;
```

则程序正确。此时，str 是指向字符数组的指针变量，str=“C Program”将字符串的首地址赋给指针变量 str，然后在 printf("%s\n", str)中输出 str 指向的字符串。

(9) 在函数内定义另一个函数而出错。如：

```
void main ()
{ ……
```

```
    int fun( int a, float b, long c )          /* 试图在主调函数内定义函数 fun */
    { …… }
    ……
}
```

应将函数 fun 定义在函数 main 之外，下面是正确的使用：

```
void main ()
{   int w, x=5;
    float y=10.5;
    long z=50000;
    int fun( int a, float b, long c );         /* 在主调函数内说明要调用的函数 fun */
    ……
    x=fun( x, y, z );                          /* 调用函数 fun */
    ……
}
int fun( int a, float b, long c )              /* 定义函数 fun */
{ …… }
```

(10) 函数的定义与引用顺序错误。如：

```
void main（）
{   float x, y;
    scanf("%f,%f",&x,&y);
    printf("The maxmum number is %f",max(x,y));
    ……
}
float max(float a,float b)
{   if (a<b)
        a=b;
    return(a);
}
```

在编译时会出错，因为在主函数中没有定义该函数，但却调用了它。正确的写法是在主函数的第一条语句前加一条声明语句：float max(float a,float b);，也可将整个函数的源程序置于主函数之前。

(11) 混淆了函数中的形参定义和函数中的局部变量定义。如：

```
f1( int a, int b )
  {     c=a;
        if(a<b)
        c=b;
        return(c);
   }
```

在编译时会出错。程序中函数的形参与局部变量应当分别定义，其中形参在函数体之前定义，而函数中的局部变量应在函数体中定义。应改为

```
f1(int a, int b)
{    int c;
       ……
 }
```

(12) 混淆数组名和指针变量的区别。

数组名中存放的是数组的首地址，但它是一个不变的地址，因此不能改变它的值，但可以将一个数组名赋给一个指针变量。例如，以下程序中的输出语句是不合法的：

```
int n, a[5]={1,2,3,4,5};
for (n=0; n<5; n++)
     printf("%d\n",*a++);
```

另外，此处输出语句使用：

```
printf("%d\n", a++);
```

同样，试图改变数组名的值也是不合法的。改正方法是：

① 用指针变量来做指针增减的操作，从而实现对数组元素的顺序操作。如：

```
int i, a[5]={1,2,3,4,5}, *p;
p = a;
for (  i=0; i<5; i++ )
       printf("%d\n",*p++);
```

② 用数组下标来指示数组元素的顺序操作。如：

```
int i,a[5]={1,2,3,4,5};
for ( i=0; i<5; i++ )
      printf("%d\n",a[i]);
```

(13) 程序中用到某些库函数时，没有用#include 命令去包含相应的头文件。如：

程序中用到了 printf、scanf、getchar、putchar 输入\输出函数，而之前没有用到

```
#include <stdio.h>
```

程序中用到了 fabs 等数学函数，但之前没有用到

```
#include <math.h>
```

(14) for 语句中漏掉“;”号。如：

```
n=10;
for( n<100; n++ )   /* 应为 for( ; n<100; n++ ) */
{    ……  }
```

(15) 非法浮点运算。

浮点运算分量不允许出现在移位运算符、按位逻辑运算符、条件（？：）、间接（*）及其他一些运算符中。当上述运算符中使用了浮点运算分量时，出现本错误。如：

```
float x, y;
……
y=x>>3;     /* 语句出错*/
```

(16) 非法指针运算。

施于指针的运算符只能是加、减、赋值、比较、间接（*）或箭头（->），且必须是同类型的指针（但不能进行加）或与常量进行加、减运算。如果和其他运算符或非指针变量运算，则出错。如：

```
int *p, i=2, a[10]={0};
p=a;
p=p+i;      /* 试图与一个非指针变量相加 */
```

(17) 此外，书写或输入程序时也容易带来大小写混淆、误用中文标点符号、变量名或函数名错漏字符、圆括号不配对、花括号不配对、/*与*/不配对、单引号或双引号不配对等问题。

例如，将输出语句：

```
printf("a=%d,b=%d,c=%d\n",a,b,c);
```

错写成下面的语句：

```
Prntf("a=%d,b=%d,c=%d\n,a,b
```

则有：① 函数名错（大写字符 P、漏字符 i）；② \n 后漏掉右双引号；③ 输出项数目只有 2 个（漏掉 c），与输出格式符数目不一致；④ 缺右圆括号；⑤ 句末无分号。

总之，只要严格按照 C 语法编写程序，输入程序时认真操作，语法错误是完全可以避免的。

4.2　常见连接错误与纠正

连接错误是指连接过程中产生的错误，主要原因是由程序中用到的标准函数和自定义函数找不到或类型不对而引起的。下面是常见的连接错误及其纠正方法。

(1) 将 C 库函数名写错。

在这种情况下，连接时将会认为此函数是用户自定义函数。此时屏幕显示:

```
Undefined symbol '<函数名>' in <程序名>
```

(2) 函数在说明和定义时类型不一致。如：

已定义一个 fun 函数，其第一行为

```
int fun(int a, float b, long c)
```

在主调函数中进行下面的声明，将出错：

```
fun(int a, float b, long c);        /* 漏写函数类型 */
float fun(int a, float b, long c); /* 函数类型不匹配 */
int fun(int a, int b, int c);       /* 参数类型不匹配 */
int fun(int a, float b);            /* 参数数目不匹配 */
int fun(int a, long c, float b);    /* 参数顺序不匹配 */
```

下面的声明是正确的：

```
int fun(int a, float b, long c);
int fun(int, float, long);          /* 函数声明可不写参数名 */
int fun(int x, float y, long z);    /* 编译时不检查函数原型中的形参名 */
```

(3) 程序调用的函数没有定义。

```
void main（）
{    float x, y;
     scanf("%f,%f",&x,&y);
     printf("The maxmum number is %f",max(x,y));
     ……
}
float max(float a,float b)
{   if (a<b)
        a = b;
    return(a);
}
```

在连接时会出错，因为在主函数中没有定义该函数但却调用了它。正确的写法是在主函数的第一条语句前加一条声明语句：float max(float a,float b);，也可将整个函数的源程序加到主函数之前。

(4) 多个文件连接时，没有在“Project/Project name”中指定项目文件（.prj 文件），此时出现找不到函数的错误。

4.3 常见运行错误与纠正

运行错误也称为逻辑错误。出现这类错误时，程序一般能编译通过，相应的源程序并无违反 C 语言语法规则，但运行结果不对。引起运行错误的原因主要有三个：一是算法错误，错误的算法导致错误的程序，从而得到错误的结果；二是由于语法与语句的使用没能正确表达出算法的本意，类似于作文中的词不达意或语不达意；三是由于 C 语言语法灵活性大，如它对变量类型的不一致性不进行语法上的严格检查，对变量和指针、数组越界没有设置屏障，某些运算（自增/自减等）还带来副作用等。因此，编程者不能过多地依赖编译程序找出错误，而要靠自己去保证程序的正确性，初学者应注意这一点。对运行错误往往需要仔细检查，充分利用调试手段进行分析才能发现。下面是常见的运行错误及其纠正方法。

(1) 使用未初始化和未赋值的变量。

```
void fun0 ( void )
{   int  i;
    float  a[10];
    char  *p;
    printf("%f, %s\n", a[i], p);
}
```

说明：非全局变量和静态变量在定义时系统不自动进行初始化，其初值为一个与该程序运行环境有关的随机数。用随机数参与运算，运算结果将不确定，如果整型随机数被用做数组下标，则将导致越界错误；如果指针类型随机数被使用，则将导致越界访问内存，可能造成运算结果错误、死机或非法操作等。

在定义变量时，所有变量都应显式地进行初始化。如果没有确定的初值，则建议数值变量初始化为 0，指针变量初始化为 NULL。

(2) 未注意数据类型的一致性。

C 语言是强类型语言，讲究数据类型不仅节省存储空间，而且有利于同类型数据之间的高效运算。但其编程对数据类型的一致性提出了苛刻的要求。初学者应该从三个方面注意变量的数据类型的一致性。

① 变量的类型定义应考虑可能的赋值。如：

```
void fun1 (void)
{   int  x, y, z;
    scanf("%d,%d", &x, &y);
    z=x*y;
    printf("%d*%d=%d\n", x, y, z);
}
```

本例若输入 x 和 y 的值使其乘积超过 32 767，根据同类型的数据进行算术运算，结果仍为同类型的数据这一规则，可知 x*y 的积必定是 int 型，则将表示不了真正的乘积值，因此变量定义时应充分考虑可能的赋值与运算结果，防止数值溢出而带来不正确的结果。

这种情况常出现在 int 型变量的定义中。一般在微机上对一个 int 型数据分配 2 个字节，因此，一个 int 型整数的范围仅为–32 768～32 767，这很容易在运算中发生溢出。上述程序在 x 和 y 的值较小时不会出问题，调试时考虑不周很容易通过。还有 int 型的表数范围与机器字

长有关。当程序从高位计算机向低位计算机移植，如从 64 位系统移植到 32 位系统时，以前从不出现的溢出问题可能会出现。

为此，编程前应预先估算运算结果的可能范围，采用相应合适的类型。如果不需要处理负数，则采用无符号类型，可扩大表数范围。如果难以预测运算结果的可能范围，则可在程序中设置溢出判断；如果超出合理的取值范围，则停止运算，转错误处理例程。

② 输入/输出的数据的格式和类型与所用格式说明符应一致。如：

```
int a;
float b;
a=3; b=4.5;
printf("%f %d\n", a, b);
```

编译时不给出出错信息，但运行结果将与原意不符，输出为

```
0.000000  16402
```

它们并不是按照赋值的规则进行转换（如把 4.5 转换成 4），而是将数据在存储单元中的形式按格式符的要求组织输出，如 b 占 4 个字节，只把最后 2 个字节中的数据按%d 格式作为整数输出。正确的输出语句应为

```
printf("%d %f\n", a, b);
```

下面的例子同样有类似的错误：

```
long sum=10; int m=10;
printf("sum=%d,m=%d\n",sum,m);
```

输出结果为

```
sum=10,m=0
```

在输出语句中，尽管 int 型变量 m 的输出格式符是正确的，但由于 long 型变量 sum 的输出格式符不对，从而影响到 m 的正确输出。正确的输出语句应为

```
printf("sum=%ld,m=%d\n",sum,m);
```

因此，①中例子的正确写法应为

```
void fun1(void)
{    long  x, y, z;
     scanf("%ld,%ld", &x, &y);
     z=x*y;
     printf("%ld*%ld=%ld\n", x, y, z);
}
```

③ 注意同类型的数据相互运算，其结果仍然是同一数据类型。因此，期望两个整数的运算自动地得出浮点数的结果是错误的，它将使运算精度受损。

如表达式 1/2*2 的值不会是 1 而是 0。又如下面的程序段：

```
float x=0.0;
int a=5, b=2;
x=a/b;
printf("x=%f\n", x);
```

printf 函数输出的结果不会是 2.5，而是 2.0。

解决的办法是，要么修改有关变量的类型，如把变量定义语句改为

```
float x=0.0, a=5.0;
int b=2;
```

要么把一个整型数强制转换为浮点数，再运算，如将原程序中有关语句改为

```
x=(float)a/b;
printf("x=%f\n", x);
```

(3) 输入变量时忘记使用地址符。如：

```
scanf("%d %d", x, y);
```

这是许多初学者刚学习 C 语言时一个常见的疏忽，或者说是习惯性的错误，因为很多高级语言的输入语句只需要写出变量名即可，而 C 语言要求用取地址符&来指明，向哪个地址标识的单元送值。因此，应写成

```
scanf("%d %d", &x, &y);
```

(4) 数据的输入格式与要求不符。

对于 scanf 函数，格式字符串中除了格式说明符外，对其他字符必须按原样输入。

假设有以下语句：

```
scanf("%d %d",&a,&b);
```

若按下面的方法输入数据：

3,4↙

则会出现输入错误。正确的输入数据间应该用空格来分隔，即输入 3　4↙。

如果 scanf 函数为

```
scanf("%d, %d",&a,&b);
```

则应按以下方法输入数据：

3, 4↙

此时，如果用“3　4”则反而错了。

(5) 在 scanf( )函数的格式说明中包含了多余信息。如：

```
scanf("input a and b:%d,%d", &a,&b );
```

本想在屏幕上显示一行信息：“input a and b:”，然后再输入 a 和 b 的值，但在 C 语言中这是不行的。如果想在屏幕上得到所需的提示信息，可以另加一个 printf 函数语句：

```
printf("input a and b:");
scanf("%d,%d", &a,&b);
```

又比如，在用 scanf 函数输入浮点数据时，不能规定小数位数。如：

```
scanf("%5.2f", &m);
```

正确的写法应为 `scanf("%f", &m);`

(6) 在 printf( )中注意“\n”的应用。

“\n”往往用于输出换行，不用或不合理的安排会使程序输出模糊，甚至造成歧义。假定连续上机运行多个程序，第一个程序执行时，末尾有一个输出：

```
printf("%d", 123);
```

第二个程序运行时，末尾有一个输出：

```
printf("%d", 456);
```

结果屏幕上显示：

```
123456
```

显然，极容易误解第二个程序执行后的结果。因此，适当插入“\n”，会使输出层次清晰，结果明了。

建议在程序的第一次输出内容前插入一个“\n”，以便与上一个程序的输出区分，在程序的最后一次输出内容后插入一个“\n”，以便与下一个程序的输出区分。

(7) 在编写复合语句时，漏掉花括号。如：

```
if(x>y)
    t=x；x=y；y=t；
```

本意是想在执行 if 语句时，若表达式（x>y）为“真”，则执行交换 x 和 y 值的操作；但由于没用花括号将复合语句括起来，因此，结果只执行 t＝x。正确的表示应该为：

```
if(x>y )
    {t=x；x=y；y=t；}
```

又如：

```
sum=0;
i=1;
while(i<=100)
    sum=sum+i;
    i++;
```

本意是实现 1＋2＋…＋100。但上面的语句只是重复了 sum＋1 的操作，而且循环永不停止，因为 i 的值始终没有改变。错误在于没有写成复合语句形式，因此 while 语句的范围到其后第一个分号为止。语句“i++”不属于循环体范围之内，应改为

```
while ( i <= 100 )
{   sum = sum + i;
    i++;
}
```

注意，采用缩格方式书写程序，并不意味着缩格的语句属于某一下属层次。C 语言是按分号来标识语句，左右花括号来标识复合语句的。缩格写法主要是便于程序员阅读程序，帮助程序员理解算法，而与程序控制结构和程序执行过程无关。

(8) 将赋值号“=”误用做关系符“==”。

例如，判断 a 与 b 的大小关系，如果相等，则输出“a equal to b”，使用下面语句：

```
if (a=b)  printf ("a equal to b");
```

假设 a＝3, b＝4，显然 a 不等于 b，按原意不应输出“a equal to b”。而实际执行该语句时，先将 b 的值赋给 a，a 的值为 4，表达式（a＝b）的值为 4。所以 if 语句中的表达式值为真（非零），因此输出“a equal to b”。

正确的语句应为

```
if (a==b)  printf ("a equal to b");
```

这种比较，对整型数是完全正确的。对浮点数可能有问题，参见运行错误(9) 。

(9) 用关系符“==”直接比较浮点数是否为 0。

判断一个运算结果是否为 0，算法上可直接用运算结果与 0 进行==关系判断。由于数据

在计算机中以二进制数形式存放，计算机对是否相等的判断采用完全按位比较的方法，因此浮点数的运算结果即使为 0，也只能是接近 0 而非完全等于 0，这样用关系符“==”直接比较 0 和浮点数，将永远无法相等。例如，对一元二次方程根判别式的判断：

```
if(b*b-4*a*c==0) {……}
```

其结果总是 0（假）!

从工程上说，绝对相等的东西是没有的，如长度，总有测量的误差。因此，比较 0 和浮点数是否相等，只要它们之间的差小于某一误差精度即可以。设误差精度为 10^{-6}，上述判断可改为

```
if(fabs(b*b-4*a*c)-0<=1e-6) {……}
```

(10) 用关系符“==”直接比较两个浮点数是否相等。

因为浮点数在计算机中的表示法和实现法的特殊性，以及浮点运算要求的精度问题，当两个数值表面相等或非常接近的浮点数用“==”比较时，结果可能不等。如：

```
float x;
scanf("%f", &x);          /* 这里输入 123.456 */
if(x==123.456)
   printf("OK");          /* 用==直接比较，不会相等，因此不会输出 */
```

表面上，这种比较没有算法问题。实际上，由于浮点常数默认是 double 类型，当 if 语句比较时，变量 x 要先做类型转换，再与常数比较，于是产生了问题。初学者编程时，很容易按算法思路直接进行浮点数是否相等的比较，以致程序不会按预想的流程执行，从而导致运行结果出错。

解决办法是，不用 float 型数据，而用 double 型数据。本例中把 x 的类型改为 double 型，可以解决此问题，但这种方法并不是最合适的。

更好的比较办法是考察两个浮点数的差的绝对值。只要该值小于某一个精度范围，就可将它们看做相等。

例如，判断浮点数 a 与 b 的大小关系。如果相等（即二者之差小于某一精度范围，设为 10^{-6}），则输出“a equal to b”，可使用下面语句：

```
if (fabs(a-b)<=1e-6)
    printf ("a equal to b");
```

(11) 在 if 语句、for 语句和 while 语句中的判断或循环控制表达式后加分号。如：

```
if（x>0）;
    x--;
```

若在 if（x>0）后加了分号，则相当于当表达式（x>0）为“真”时，执行了一条空语句，此时不论 x 是否大于 0 都将执行语句 x--。又如：

```
for（i=0; i<10; i++）;
    printf（"%d\n"，a[i]）;
```

本希望依次输出 a[i]的 10 个值，但由于 for 语句后加了一个分号，使循环体成了空语句，因此只能输出一个 a[10]的值（for 循环后 i＝10）。

由于分号表示循环和判断语句的终结，因此后面的代码都不算循环体或分支，而是与循环和判断平行的代码。这种情况使得程序执行流程与预期不符。即使在没有循环体的情况下，这也使得程序的可读性下降。错误只由一个字符引起，很难发现。

解决办法是，禁止在循环判断语句末尾出现分号。在循环体为空的情况下，使用下面形式代码：

```
while(……){}
```

这样，只要在循环判断语句末尾出现分号，就肯定是错误的输入。

(12) switch 语句的各分支中漏写 break 语句。如：

```
scanf("%d",&x);
switch(x)
{   case 1: printf("Monday\n");
    case 2:     printf("tuseday\n");
    case 3: printf("wednsday\n");
    case 4: printf("thurthday\n");
    case 5:     printf("friday\n");
    case 6: printf("saturday\n");
    case 7: printf("sunday\n");
    default:    printf("error\n");
}
```

本意是根据输入的数字输出对应的星期数，但当 x 的值为 5 时，将连续输出：

```
friday
saturday
sunday
error
```

显然，输出结果与原意不符。这是由于在各 case 分支中未写 break 语句。应改为在每个 case 分支的 printf 语句之后增加一条 break 语句，使每次执行完一个 case 分支后，立即终止 switch 语句的执行。

(13) switch 语句没有 default 分支。

使用 switch 语句时，因为所有可能的情况都有相应的 case 处理，所以省略了 default 分支。如果因为某种没有预想到的原因，而产生没有对应的 case 可以处理的情况，那么程序的继续运行可能会出现各种非预期的情况。

建议永远都要提供 default 分支。如果 default 中确实没有什么可做的，就提示错误发生，并输出程序当前状态信息，以便捕获错误，利于调试。

(14) break 语句和 continue 语句的误用。

continue 语句是结束这一次循环而执行下一次的循环。break 语句是跳出整个循环而执行循环之后的语句。如：

```
void main()
{   int n;
    for (n=100; n<200; n++)
        {    if(n%3==0)
                 break;     /* 改为 continue; */
             printf("%d\n",n);
        }
}
```

本意是要输出 100 至 200 之间不能被 3 整除的数，但由于程序中使用了 break 语句，因而造成当出现一个能被 3 整除的 n 时，就跳出循环体，而不判断下一个数。应改用 continue 语句来替代 break，使得当出现一个能被 3 整除的 n 时，只结束执行本次循环，而不是跳出整个循环。

(15) 在定义数组时，将定义的“元素个数”误认为是“数组最大下标值”。如：

```
main()
{   static  int a[10]={1,2,3,4,5,6,7,8,9,10};
    int  i;
    for(i=1; i <= 10; i++)
    printf("%d", a[i]);
}
```

本意是想输出 a[1]到 a[10]，但在数组初始化时只给 a［0］到 a［9］赋值，并未对 a[10]赋值。这是一些初学者常犯的错误。C 语言规定：定义时用 a[10]，表示 a 数组有 10 个元素，而不是可以用的最大下标值为 10。

应当把 for 语句改为 for（i=0；i<=9; i++）。

(16) 数组下标越界。

一个数组若定义了 n 个元素，则合法的下标为 0 到 n–1。如果程序中出现超出最大下标的数组元素，则越界的下标将访问数组以外的空间，会影响其他变量甚至程序执行，还可能导致安全漏洞或非法操作。

解决办法是，养成从 0 开始计数的习惯，所有使用下标的地方都仔细估算是否产生越界。必要时，可定义比需求多的数组元素（如需要 100 个数组元素，就定义 120 个），或者在数组的前后各定义一个永不使用的数组，此法有时可以从表面上解决问题，但只能是权宜之计。

(17) 字符串没有‘\0’终结符。

C 语言规定所有与字符串有关的库函数都假定字符串以‘\0’结尾。如果预设的字符串没有终结符‘\0’，程序将越界访存，可能得到非预期的数据，也可能覆盖了不该覆盖的数据。因此，程序员应深刻理解各种字符串相关的库函数，所有的字符串都要具有终结符‘\0’。

(18) 使用++或--运算符时出现的错误。如：

```
int n=10;
printf("%d\n", n++);
```

有人认为执行以上 printf 语句时，将输出 11，而实际上输出 10。这是由于 n++表示先用 n 的值参加运算，然后再使 n 的值增 1，而++n 则先使 n 值增 1，再进行输出。这样，执行以上 printf 语句时，将输出 10。因此，编程时应仔细区分前缀++与后缀++、前缀--与后缀--的区别。又如：

```
void main()
{   int *p, a[5]={1, 3, 5, 7, 9};
    p=a;
    printf("%d", *p++);
}
```

有人认为*p++的作用是先使 p 加 1，即指向 a[1]处，然后输出 a[1]的值 3。实际上，按照 C 运算规则，一元运算顺序是从右至左进行，即先执行 p++，后运算*。而 p++的作用是先用 p 的原值，用完后再使 p 加 1，因此上述程序输出的是 a[0]的值 1。

使用++或--运算符时，一定要避免歧义。如果无把握，宁可多加括号。如上面的*p++可改为*(p++)。

(19) 试图让形参值的变化使实参的值也发生变化。如：

```
void main()
{   void swap();
    int a = 3,b = 4;
```

```
    swap(a,b);
    printf("%d %d\n",a,b);
}
    void swap(int x,int y)
{   int t;
    t = x; x = y; y = t;
}
```

原意是通过调用 swap 函数使 a 和 b 的值对换，然后在 main 函数中输出已交换了值的 a 和 b。但上述程序达不到这一目的。因为 C 语言中参数的数据传递是“值传递”，即单向传递，只能由实参传给形参。如果想从函数得到一个变化了的值，则应该用指针变量。

(20) 不定义函数参数或返回值的类型。如：

```
fun00()                          /* 无函数类型和函数参数 */
{
}
void func(void)
{   printf("%d", fun00());       /* 会得到返回值，但值不确定 */
}
```

若不定义函数类型，编译器将其处理为 int 型。若不定义函数参数，可以有任意个 int 类型参数。调用者提供的所有参数，该函数都得不到。一般设计这样函数的目的是定义一个没有参数和返回值的函数，所以不会用 return 指明返回值，导致调用者得到无意义的返回值，它会引起歧义，降低可读性。一般应指明函数类型和函数参数。若不定义，应指明是 void 型。

(21) 有返回值的函数不用 return 指明返回值。如：

```
int func(void)
{   return;     /* 未指明返回值，有无 return 效果都一样 */
}
```

若定义了函数类型，应该用 return 指明返回值，否则导致调用者得到无意义的返回值。应规定所有的返回值的函数都必须指明返回值。即使无值可返回，也可以返回标志。例如，运算顺利返回 1，不顺利返回 0。

(22) 调用函数后，不检查函数是否正确执行。

malloc()、open()、scanf()等函数执行时出错率较高。在调用这些函数时，应检查返回值是否错误。如果该函数已经出错，调用者在错误状态下把错误的结果用来进行计算，必然错上加错。因此，只要调用有出错可能的函数，就必须检查是否出错。

如将文件打开语句由

```
fp=fopen("filename1.dat", "r");
```

改为

```
fp=fopen("filename1.dat", "r");
if(fp==NULL)
{   printf("cannot open file");
    exit(0);
}
```

(23) 指针的概念模糊，对指针的定义与引用分不清。如：

```
char *p, n;
*p=&n;
```

是不合法的。因为说明语句中 p 前面的*表示该变量是一个指针变量，而执行语句中 p 前面的*号是一个指针运算符，*p 表示的是 p 指针所指向的存储单元的内容。因此，应将以上赋值语句改为*p=n;或者 p=&n;。

又如，在以下程序中：

```
int n=0, *s=&n;
float *q;
s=q;
```

赋值语句 s=q 是非法的。

不同类型的指针之间的赋值必须通过强制类型转换来实现。把以上赋值语句改为 s=(int*)q;。又如：

```
char *p;
*p=(char *)malloc(100);
```

第二条语句很可能使所在整个程序瘫痪。因为 malloc()返回的地址没有被送到指针 p 中，而是送到了 p 指向的内存位置，结果还不清楚 p 指向哪一个地方。正确的语句应为

```
p=(char *)malloc(100);
```

(24) 返回指向局部变量的指针。如：

```
int *func()
{   int n=0;          /* n 是局部变量 */
    return &n;        /* 返回局部变量的地址 */
}
```

该指针函数把局部变量的地址以返回值等形式提交给调用者。由于该变量的空间已经随着函数的退出而释放，该空间可能已经禁止访问或被分做它用，因此调用者引用此指针是危险的、无意义的。

解决办法是，直接返回数值而不是指针。如果数据量大，应该由调用者提供存储空间，通过参数传递指针给函数。

(25) 随意修改全局变量的值。

按照 C 语法规则，任何模块、函数都可随意修改全局变量的值。如果随意修改全局变量的值，则很难跟踪目前的改变是由谁做的，说明了什么问题，在不恰当的改变时机改变它，会引起混乱，并使模块之间造成强耦合。

解决办法是，把全局变量以静态方式封装在模块内，只提供读的函数，不提供写的函数，或者只提供受限的写函数。

(26) 路径名错误。

在 DOS 中, 斜杠（\）表示一个目录名或分隔符，而在 C 语言中斜杠是某个字符串的一个转义字符，这样，在用 C 语言的字符串说明一个路径名时应考虑 "\" 的转义的作用。例如，有这样一条语句：

```
file=fopen ("c:\new\tbc.dat", "rb");
```

目的是打开 C 盘 new 目录中的 tbc.dat 文件，但实际做不到。这里 "\" 后面紧接的分别是 "n" 及 "t"，而 "\n" 及 "\t" 在 C 语言中表示换行及 Tab 跳格字符，因此运行时将认为它是不正确的文件名。正确的写法应为

```
file=fopen ("c:\\new\\tbc.dat", "rb");
```

这里用双写的“\\”表示一个斜杠字符。

(27) 使用文件时忘记打开，或打开方式与使用情况不匹配。

如对文件的读写，首先必须打开相应的文件。如果用只读方式打开，却企图向该文件输出数据，则会出错，如：

```
FILE *fp;
char ch;
fp=fopen("test.txt","r");   /* 指明用只读方式打开文件 */
ch=getchar();
while (ch !='#')
{   ch=ch+4;
    fputc(ch,fp);           /* 试图对 fp 写入数据 */
    ch=getchar();
}
```

对以“r”方式（只读方式）打开的文件，却试图用 fputc()写入数据操作，显然在运行时是达不到目的的。应改为

```
fp=fopen("test.txt","w");
```

(28) 文件打开后不主动关闭。

文件使用完毕后，应养成立刻关闭的编程好习惯。有的程序常忘记关闭文件，虽然系统在程序退出时会自动关闭所有文件，但文件使用完毕后不立刻关闭，会影响其他程序使用此文件。如果在打开状态下死机或者掉电，则可能造成数据丢失，极端严重情况下将影响到整个硬盘。

(29) 忘记对 malloc()返回值进行检查。

当内存耗尽时，malloc()返回 NULL 值，此时如果继续使用 malloc()返回的指针值是非法的，极可能造成系统崩溃。下面是一个正确的程序，其中增加了指针合法性的检查。

```
#include    <stdio.h>
int main(void)
{   char *p;
    p=(char *)malloc(100);
    if(!p)              /* 若 p 值为 NULL */
    {   printf("Out of memory.\n");
        exit(1);        /* 退出，返回操作系统 */
    }
    gets(p);
    printf(p);
    free(p);
    return(0);
}
```

(30) 用 malloc()申请的内存不用 free()。

如果从 malloc()申请到内存开始，一直到不再使用此内存区域，甚至到程序退出，都不用 free()，则有可能造成内存浪费和内存泄漏。应养成 malloc()和 free()成对使用的习惯。

(31) 使用已经被 free()的指针。

内存用 free()释放后，如果还通过指向这块区域的指针访问它，那么由于此块内存区域

可能已被分做它用，因此可能得到不确定的数据或造成非法操作。

解决办法是，指针被 free()释放后，把它赋值为 NULL。在使用指针前，通过 if 判断或 assert 断言来避免对 NULL 的引用。

(32) 成对函数不在同一个模块或函数内调用。

open()和 close()，malloc()和 free()这样必须成对使用的函数称为成对函数，一般在申请函数和释放函数时匹配。如果成对函数不在同一个模块或函数内调用，则会造成程序整体结构混乱，条理不清晰，有不释放或重复释放的可能。因此，配套的成对函数调用只能出现在一个模块或函数内。编程时，写下申请函数的同时，马上写好释放函数，然后再完善中间的代码。

(33) 希望“按任一键返回”而错用 getchar()。

getchar()的功能是从键盘读一字符，并等待回车。如果算法希望用户“按任一键返回”或者用户只需按一键即程序自动切换显示内容时，则使用 getchar()将造成用户按任一键后系统仍然处于等待状态。正确的做法是使用 getch()，该函数的功能是从键盘读一字符，且不回显，也不等待回车。

对多数编译程序而言，getch()的原型是在 conio.h 中建立的，因此编译预处理要包含该头文件。下面是其应用的一段正确的程序：

```
#include    <conio.h>
……
printf("Press anykey continue……\n");
getch();
……
```

在 VC++中，getch()前有一下画线，即_ getch()。

4.4 常见操作错误与纠正

操作错误是指上机过程本身带来的错误，致使程序不能正常地编辑、编译、连接和调试运行。下面是常见的操作错误及其纠正方法。

(1) 开发环境配置不当

Turbo C 在安装时，自动设置了一个内部默认的程序开发环境，即对编辑程序、编译程序、连接程序等的工作状态和环境变量给定了初始值，并保存在一个特定的名为 TCCONFIG.TC 的配置文件中。该配置文件位于 TC 子目录下。TC 启动后，自动将该文件调入内存并遵循其配置。如果实际使用的包含文件所在的目录、库函数等文件所在的目录和 C 编译程序所在的目录与 Options 菜单中 Directories 项的设置值不一致的话，连接时往往出现：

```
Unable to open input file '包含文件名'
```

或

```
Unable to open input file 'COS.OBJ'
```

此时，应检查 Directories 项的设置值。

建议上机操作前，首先设置好 Directories 项。

(2) 开发环境进行了配置，但未保存。

开发环境配置好后，一定要选择 Options 菜单下的 Save options 项，将程序开发环境保存

在 TCCONFIG.TC 的配置文件中。如果只对 Options 菜单中的 Directories 项进行了配置而不保存，则所进行的配置是无用的，配置不当的故障会依然存在。

(3) 未注意编译连接的优先顺序。

如果执行程序后显示的结果与程序指定的形式大不相同，如要求显示“x=”的字样而出现“a=”的字样，则往往是机器运行了另一个程序，而非当前的源程序。原来 TC 编译程序对文件的处理有一个优先顺序，依次是 Project name 中的.prj 文件→Primary C file 中的.c 文件→最后装入编辑窗口中的文件。碰到这种情况，一定要看看 Project→Project name 项或者 Compile→Primary C file 项，否则，有可能编译连接执行的不是当前窗口中的程序。

(4) 用 noname 作为文件名。

noname.c 是系统默认的源文件名。如果企图省事不给出用户自己的文件名，则有可能与前一次采用 noname.c 的源文件混淆，而出现与本程序不相符的执行结果。这种情况常发生在当前源程序有错误，而又存在已经编译连接成功的 noname 文件时。建议每输入编辑 1 个程序，存盘时一定给出用户自己的文件名，这是我们应有的良好的上机操作习惯。

第 4 部分　C语言程序设计实验

实验 1　C语言上机初步

一、实验目的

1．熟悉 C 程序从编辑、编译、连接到运行，得出运行结果的过程。

2．掌握 C 程序的特点。

3．初步了解 VC 环境下单步跟踪程序的执行过程。

二、实验准备

1．复习本书第 3 部分第 1 章内容，了解 C 语言程序的开发过程。

2．复习教材《C 程序设计与应用教程》第 1 章的内容，掌握 C 程序的一些特点：函数是 C 程序的基本组成单位；每个源程序有且仅有一个主函数 main；每个程序基本上都有 printf( )、scanf( )函数的调用；每条语句后面必须有一个分号；C 程序书写自由，一行内可写几条语句；/*……*/是 C 程序的注释；C 程序严格区分大小写，等等。

3．编程：实现在屏幕上显示如下三行文字：

```
Hello, the world !
Welcome to the C language world!
Everyone has been waiting for.
```

三、实验内容

1．在某个盘上建立一个自己的文件夹。例如，学生何小兵，可以为自己的文件夹取名为 hxb，后面的实验文件都保存在该文件夹下。

2．新建一个 Source File，文件名是 first.c，然后输入下列程序。

```
#include <stdio.h>
void  main()
{   printf("C Programe is very fun.");
    printf("I will work hard to master it!");
}
```

将 first.c 编译、连接后运行。运行时，观察其输出结果是一行还是两行？如果要使输出结果为两行，则应如何修改源程序？

3．新建一个 Source File，文件名是 second.c，然后输入下列程序。

```
#include <stdio.h>
void  main()
{   printf("  *\n");   print("  ***\n");
    printf("*****\n")
    printf("*******\n");
}
```

该程序有多处编译错误，请将编译时的出错信息记录下来，并将错误更正后重新编译。

相关知识：C 语言严格区分大小写；printf()是输出函数；C 语言的语句结束标记是“;”。

4．将实验准备中编写好的程序 1 上机输入、编译、连接并运行。

5．新建一个 Source File，文件名是 area.c，然后输入下列代码。

```
#include  "stdio.h"
void main()
{                          /* 以下是说明语句部分 */
    int  a,b,h;            /* 变量说明，设梯形的上底、下底和高为整型变量(int 型变量) */
    float  area;           /* 变量说明，设梯形面积 area 为实型变量(float 型变量) */
                           /* 以下为执行语句部分 */
    printf("Input a,b,h:");
    scanf("%d,%d,%d",&a,&b,&h);
                           /* 调用输入函数 scanf()，从键盘分别输入数据给 a,b,h */
    area=(a+b)*h/2;
    printf("area=%f\n",area);
                           /* 计算出梯形面积，将结果赋给变量 area */
}
```

将程序编译、连接后，利用 F10 单步跟踪程序的运行，注意观察程序的执行流程。

四、实验报告要求

1．结合使用的 C 语言程序开发平台说明 C 程序的开发过程。

2．结合上机的体会，列举出 C 程序的特点。

3．回答实验内容第 2 题的问题：如果要使输出的结果为两行，应如何修改源程序？

4．将实验内容第 3 题中的编译错误记下来，译成中文，并将正确程序写出来。

5．列出所编程序 1 的清单并给每行加上注释说明。

实验 2　数据类型与表达式

一、实验目的

1．掌握 C 语言基本的数据类型，熟悉如何定义整型、字符型或实型变量，以及对它们赋值，了解以上类型数据输出时所用的格式说明符。

2．掌握 C 语言输入输出函数的使用：scanf()，printf()，getchar()，putchar()。

3．掌握算术运算符、赋值运算符，以及各运算符的优先级问题。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 2 章的内容：

① printf()、scanf()、getchar()、putchar()等函数的使用方法；

② 格式说明符；

③ 转义字符；

④ 常量和变量的区别和联系；

⑤ 算术运算符、赋值运算符。

2．填空题：以下程序实现两个变量值的交换，请填空，实现该功能。

```
#include <stdio.h>
void  main( )
{   int x,y,z;
    x=10; y=15;
    printf("before  swap  x=%d,y=%d\n",x,y);          /* 交换之前 x,y 的值 */
    (1)    ;   (2)    ;   (3)    ;
    printf("after  swap  x=%d,y=%d", x, y);           /* 交换之后 x,y 的值 */
}
```

相关知识：借助中间变量交换两个变量的值。

3．验证题 1：阅读如下程序，写出预习结果。

```
#include <stdio.h>
void  main( )
{   char  c1,c2;
    c1=97;  c2='a';
    printf("c1=%c  c1=%d\n", c1,c1);   /*%c 是输出字符,%d 是输出字符的 ASCII 值 */
    printf("c2=%c  c2=%d\n", c2,c2);
}
```

相关知识：字符型变量可以看成一个小整数。字符型变量的值能够以两种形式输出：一是以字符的形式输出（%c）；一是以整型量形式输出其十进制的 ASCII 码值（%d）。

4．验证题 2：预习下列程序，写出预习结果。

```
#include  <stdio.h>
void  main()
{   char  c1='a', c2='b', c3='c';
    char  c4='\101', c5='\116';              /* 用八进制 ASCII 码值表示的字符 */
    char  c6='\x30',c7='\x41',c8='\x61';     /* 用十六进制 ASCII 码值表示的字符 */
    printf("H\tI\b\bJ   k" );                /* '\t'是制表符，'\b'是退格符 */
    printf("a%cb%c\t c%c\t abc\n", c1, c2, c3);
    printf("c4=%c  c5=%c\n", c4, c5);
    printf("c6=%c  c7=%c   c8=%c", c6, c7,c8);
}
```

相关知识：

(1) 转义字符，分成两类：

① 一类是字符转义字符，由一个反斜杠加上一个字符或符号组成，例如：\a、\n、\t、\v、\b、\r、\f、\\、\’、\”；

② 另一类是数字转义字符，又分为八进制转义字符和十六进制转义字符。前者由反斜杠“\”和随后的 1～3 个八进制数字构成，后者由反斜杠“\”和字母“x”（此处 x 为小写，不能是大写的 X）及随后的 1～2 个十六进制数构成。实际上，ASCII 码表的任何字符都可以用八进制转义字符或十六进制转义字符表示。

(2) 同一个字符，可能有多种表示方法，如字符 A 就可表示为‘A’、‘\101’、‘\x41’等。

5．验证题 3：下列程序计算 $y=\frac{1}{2}x^3-\frac{3}{5}x^2+2.5x-1.25$，阅读程序，写出预习结果。

```
#include <stdio.h>
void main()

{   float x,y;
```

```
    x=4.0;
    y=1/2*x*x*x-3/5*x*x+2.5x-1.25;          /* 注意"/"运算符的特点 */
    printf("%f",y);
}
```

相关知识：“/”运算符的特点：它是个二元运算符；若左右操作数都是整型量，结果是整除；若左右操作数中只要有一个是实型量，则结果是浮点数。

6．验证题 4：预习如下程序，写出预习结果。

```
#include <stdio.h>
void main()
{   int  x, y, z;
    x=4; y=5; z=0;
    x+=y+=z;                                /* "+="运算符是自左向右结合的 */
    printf("x=%d,y=%d\n", x, y);
}
```

7．编写程序 1：图书都有一个 ISBN 标识，例如 978-7-302-15970-4，其中 978 代表图书产品；7 代表地区号，表示中国；302 代表出版社代码；15970 代表书序码；4 代表计算机校验码。编写一个程序来分解用户输入的 ISBN。格式如下：

输入格式：　ISBN：978-7-302-15970-4↙

输出格式：　图书产品号：978
　　地区号：7
　　出版社：302
　　书序号：15970
　　计算机校验码：4

提示：scanf("%d-%d-%d-%d-%d", &a,&b,&c,&d,&e);

8．编写程序 2：编程实现一个三位数的反向显示。程序的输入输出格式要求如下：

请输入一个 3 位整数：257↙

反向显示的结果是：752

提示：有两种解题思路，选择其中之一实现。

(1) 读入一个整数：scanf("%3d", &n)，然后构造相应的算术表达式将 n 的 3 个位剥离开来：n%10 得 n 的个位，(n-n/100*100)/10 得 n 的十位，n/100 得 n 的百位。

(2) 分成 3 位来读数：scanf("%1d%1d%1d", &a,&b,&c)，再按 c,b,a 的顺序输出数据。

9．对以上各程序准备好测试数据。

三、实验内容

1．改错题：要求下列程序能按指定的格式输入/输出数据。该程序有多处错误，请对该程序进行相应的修改。

```
#include <stdio.h>
void    main ()
{   float a,b,c,s,v;
    printf(输入三个小数:);    /* 要加上双引号 */
    scanf("%d%d%d",a,b,c);  /*输入项是地址量，还要注意不同的数据类型用不同的格式说明符 */
    s=a*b;
```

```
    v=a*b*c;
    printf("%d   %d   %d",a,b,c);
    printf("s=%f\n",s,"v=%f\n",v);  /* printf 函数的调用格式不对，复习其调用格式 */

}
```

当程序执行时，输入输出数据的格式要求如下：

输入三个小数:1.0　2.0　3.0↙

```
a=1.000000,b=2.000000,c=3.000000
s=2.000000,v=6.000000
```

相关知识:

(1) 不同的数据类型需用不同的格式说明符，例如，int 型对应于%d、float 型对应于%f、char 型对应于%c 等。

(2) scanf()函数要求输入项是一个地址量。

(3) printf()函数的第一个参数，是用双引号括起来的字符串。在这个字符串参数中有 3 种类型的数据：一是普通字符，原样输出；二是格式控制符，也称为占位符，指示后面的数据按什么格式输出；三是转义字符，按其字符的控制功能进行输出控制。

2．将填空题输入、编译、连接并运行。

3．将验证题 1 输入、编译、连接并运行，验证预习结果。

4．将验证题 2 输入、编译、连接并运行，验证预习结果，并思考：若要在输出数据之前加一个提示信息，例如“y=”，应该如何实现？

5．将验证题 3 输入并编译。该程序有编译错误，记下出错信息。改错后重新编译、连接并运行，运行结果正确吗？如不正确，应如何修改？修改后，重新编译、连接并运行，验证运行结果，并注意观察：用%c 输出字符时，字符的定界符（单引号）会自动输出吗？

6．将验证题 4 输入、编译、连接并运行，验证预习结果。

7．将编写好的程序 1 输入、编译、连接并运行，运行多次，用不同的数据测试并记录输出结果。

8．将编写好的程序 2 输入、编译、连接并运行，运行多次，用不同的数据测试并记录输出结果。

四、实验报告要求

1．将修改后的改错题程序清单列出来，并加上注释。

2．将填空题的程序清单列出来。

3．回答实验内容第 3 题的问题：用%c 输出字符时，字符的定界符（单引号）会自动输出吗？

4．将验证题 3 编译时的出错信息写下来，且译成中文，该编译错误对程序的运行有何影响？写出得到正确运行结果的修改后的程序，并分析原程序运行结果不正确的原因。

5．分析验证题 4 的运行结果。

6．将上机运行后的程序 1 清单列出来，并加上必要的注释。

7．将上机运行后的程序 2 清单列出来，并加上必要的注释说明。

实验 3　选 择 结 构

一、实验目的

1．掌握 C 语言表示逻辑量的方法（以 0 代表“假”，以非 0 代表“真”）。
2．学会正确使用关系表达式、逻辑表达式及条件表达式。
3．熟练掌握 if 语句和 switch 语句。
4．掌握一些简单的算法：求 4 个数的最值、对 4 个数进行排序、分区间求值等。
5．进一步练习 VC 环境下调试工具的使用。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 3 章的下述内容：
(1) 关系表达式、逻辑表达式；
(2) if 语句的使用；
(3) switch 语句的使用。
2．阅读本书第 3 部分 2.2 节或者 3.4 节，熟悉单步跟踪调试，了解功能键 F10 的使用。
3．验证题 1：预习下列程序，写出预习结果。

```
#include <stdio.h>
void main( )
{   int  a,b,c;
    a=1;b=2;c=3;
    printf("%d\n",b>c?b++:c++);                  /* 条件运算符 */
    printf("%d\n",b>c?b++:c++);                  /* 条件运算符 */
    a<b?printf("%d\n",c):printf("%d",b);         /* 条件运算符 */
}
```

4．验证题 2：预习下列程序，写出预习结果。

```
#include  <stdio.h>
void  main()
{   int  x=100,a=10,b=20;
    int  v1=5,v2=0;
    if(a<b)                  /* 如果 a 小于 b，将执行下列操作 */
        if(b!=15)            /* 如果 b 不等于 15，将执行下列操作 */
            if(!v1)          /* 第 7 行 */
                x=1;
            else if(v2)      /* 第 10 行 */
                x=10;
            else
                x=50;
    printf("x=%d",x);
}
```

5．编写程序 1：实现求 4 个整数中的最大值。

提示：首先，假设第一个数据最大 max，然后用这个 max 和其他数据逐个比较，直到把最大数据找出来。算法流程如图4.1所示。

6．编写程序 2：编程实现输入 4 个整数，按升序输出这 4 个数。

提示：采用比较排序算法，即用一系列的比较来实现排序。假设这 4 个数分别是 a, b, c, d，将 a 和 b, c, d 三个数逐个比较，若 a 大于其中的任一个数，则通过中间变量 t 交换 a 和这个数，使得 a 保存这 4 个数中最小的一个；然后再将 b 和 c, d 两个数逐个比较，使得 b 存放剩下 3 个数中最小的一个；然后再将 c 和 d 进行比较，使得 c 存放这两个数中小的一个，d 存放大的一个。这样，经过 6 次 if 语句的比较，即可实现这 4 个数的升序排列。比较排序算法的流程如图 4.2 所示。

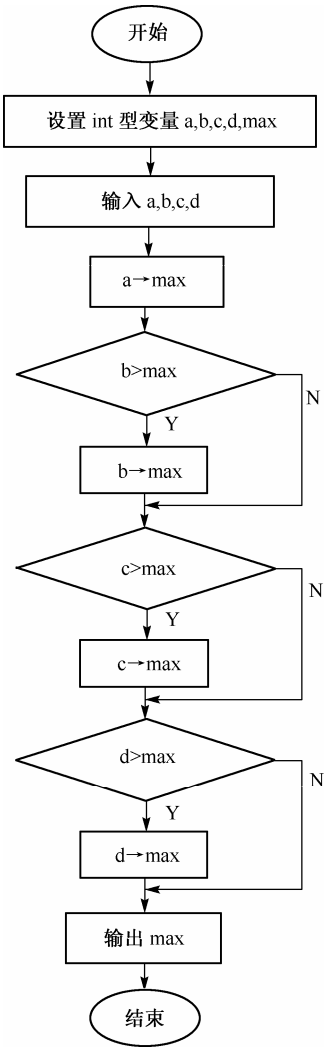

图 4.1 程序 1 流程图

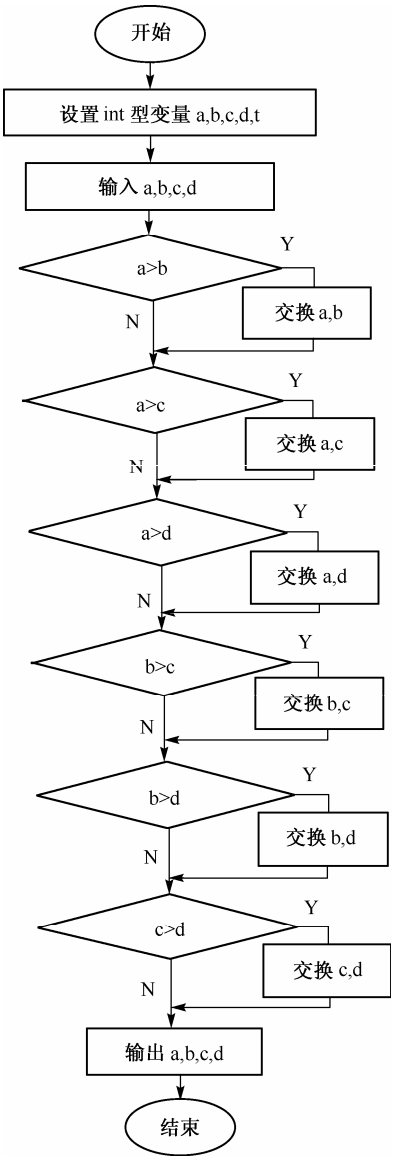

图 4.2 程序 2 流程图

7．编写程序 3：实现下面的分段函数

$$y=\begin{cases} x & (x<1) \\ 2x-1 & (1\leq x\leq 10) \\ 3x-11 & (x>10) \end{cases}$$

相关知识:

(1) 用 if-else if 实现分段函数;

(2) 类似于 1≤x≤10 这样的数学区间，在 C 语言中的实现是：1<=x && x<=10。

8．验证题 3：阅读下列程序，思考若输入 4，该程序的结果是多少？

```
#include <stdio.h>
void main( )
{   int  day;
    scanf("%d", &day);
    switch (day)
    {   case  1 :   printf(" Monday ,\n");
        case  2 :   printf(" Tuesday ,\n");
        case  3 :   printf(" Wednesday ,\n");
        case  4 :   printf(" Thursday , \n");
        case  5 :   printf(" Friday ,\n");
        case  6 :   printf(" Saturday,\n");
        case  7 :   printf(" Sunday ,\n");
    }
}
```

相关知识：break 语句的使用。

9．对以上各程序准备好测试数据。

三、实验内容

1．改错题：以下程序实现若 x 和 y 相等，则输出 Yes，否则输出 No。输入几组不同的数据，如一组是 4 和 5，一组是 6 和 6，一组是 0 和 0，测试该程序的运行情况。程序有错吗？若有，则改正。

```
#include  <stdio.h>
void  main()
{   int  x,y;
    scanf("%d%d", &x,&y);
    if(x=y)  printf("Yes");
    else  printf("No");
}
```

相关知识：“=”和“==”代表了不同的运算符。“=”是赋值运算符；“==”才是相等的关系运算符。

2．将验证题 1 输入、编译、连接并运行，验证预习结果。

3．将验证题 2 输入、编译、连接并运行，验证预习结果。再用 F10 单步跟踪该程序的运行，仔细观察 if 的执行路径。

4．将编写好的程序 1 输入、编译、连接并运行，用不同的数据测试其执行情况。

5．将编写好的程序 2 输入、编译、连接并运行，用不同的数据测试其执行情况。

6．将编写好的程序 3 输入、编译、连接并运行，用不同的数据测试其执行情况。

7．将验证题 3 输入、编译、连接并运行。用 F10 单步跟踪程序的运行，在跟踪时，输入 4 测试运行情况，并思考：若输入 4 时，只输出 Thurthday，应该如何修改程序？

四、实验报告要求

1．分析改错题的错误原因。由于这类错误不是语法问题，因此没有编译错误。但程序运行时结果不对，这类错误属于逻辑错误。

2．分析验证题 1 的执行结果。

3．分析验证题 2 的执行结果，并回答验证题 2 中第 7 行的“if(!v1)”和第 9 行的“if(v2)”分别代表什么？

4．列出上机运行后的程序 1 清单，并回答若要实现求最小值，应如何修改程序？

5．列出上机运行后的程序2清单，并回答若想改为由大到小的排序（即降序），应如何修改程序？

6．列出上机运行后的程序 3 清单，并画出程序流程图。

7．回答“实验准备”第 8 项的思考问题。

实验 4　循 环 结 构

一、实验目的

1．掌握 for 语句、while 语句和 do～while 语句实现循环的语法格式和方法。

2．掌握嵌套循环结构的执行过程。

3．掌握 continue 及 break 在循环结构中的作用与区别。

4．掌握用循环实现的常用算法（求一批数据中的最大值、最小值，求平均值，统计个数，等等）。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 3 章的内容：

① 循环的三种实现方式：for 循环、while 循环和 do～while 循环；

② 循环的嵌套；

③ continue 和 break 的作用与区别。

2．阅读本书第 3 部分 2.2 节或者 3.4 节，进一步熟悉单步跟踪调试的内容。

3．验证题：阅读下列程序，写出预习结果。

```
#include <stdio.h>
void main( )
{   int i,j,x;
    for(i=0,x=0;i<2;i++)
    {   x++;
        for(j=0 ; j<3;  j++)
        {   if(j%2)  continue;
            x++;
        }
        x++;
    }
    printf("x=%d\n", x);
}
```

4．改错题：下列程序的功能是实现输入一批考试分数（以–1作为结束标志），若输入大于 100 分，则提示重新输入，然后计算最高分、最低分与平均分。程序有错误，请预习并改正。

```
#include <stdio.h>
void main()
{   int mark;
    int n=0 ;   sum=0;          /* 定义变量有错 */
    int max=100 ;  min=0;       /* 定义变量有错，且变量的初始化有错 */
    for(; ;) ;                  /* 这是一个死循环 */
    {   scanf("%d", &mark);
        if(mark>100)
        {   printf("Mark>100, Please reinput \n ");
            break;              /* 题意要求重新输入而不是退出循环 */
        }
        if(mark=-1)             /* 注意"="和"=="的不同 */
            break;
        n++;
        sum=sum+mark;
        if(mark>max)
            max=mark;
        if(mark<min)
            min=mark;
    }
    sum=sum/n;
    printf("max=%d, min=%d, aver=%d \n", max, min, sum);
}
```

提示：该程序有多处错误，下面给出几处错误的说明，根据这些提示，修改源程序。

(1) 当求一批数的最大值时，若已知这一批数的取值范围[a, b]，则将范围的下限 a 作为存放最大值的变量的初始值；当求最小值时，情况相反，即将范围的上限 b 作为存放最小值的变量的初始值。本例分数的取值范围为[0, 100]，求这一批分数的最高分时，最高分 max 的初始值应该设为 0，然后在程序循环中逐渐地升高，直至求出真正的最高分；求这批分数的最低分 min 时，初始值应设为 100，然后在程序循环中逐渐地降低，直至求出真正的最低分。

(2) 实现“若输入大于 100 分，则提示重新输入”功能的语句有错。

(3) 实现“用–1 作为结束标志”功能的语句有错。

(4) sum 变量和 min 变量没有定义。

(5) 注意，for 圆括号后面有一个分号，将会造成死循环。

5．编写程序 1：输入 20 个 1～90 的整数，分类统计其中 1～30、31～60、61～90 的数各有多少个？

提示：

(1) 设计 3 个计数变量 count1, count2, count3，其中 count1 计 1～30 的个数，count2 计 31～60 的个数，count3 计 61～90 的个数。

(2) 用 for 循环 20 次实现：输入一个整数 x，判断该数所在的范围，若 x 是 1～30，则 count1 加 1；若 x 是 31～60，则 count2 加 1；若 x 是 61～90，则 count3 加 1。

(3) 最后，输出统计的结果。

6．编写程序 2：编程实现输入一个不超过 5 位数的整数，然后反向显示。（参考实验 2 程序 2）

提示：该算法实现的流程如图4.3 所示。

7．编写程序 3：设计一个程序，实现 1!+2!+...+n!，其中 n 值由键盘输入，且是一位整数。

8．编写程序 4：输出下列图形。

```
    *
   ***
  *****
 *******
*********
```

提示：考虑两点，一是每行第一个星号的输出位置，一是每行星号的个数。

(1) 每行第一个*号输出位置比上一行少一个空格，先定位好第一个*的输出位置：

```
for(j=1;j<=40-i;j++)  printf("%c",' ');     /* 其中 i 是行号 */
```

(2) 每一行的星号与行号 i 有关，即每一行输出 2*i–1 个*号：

```
for(j=1;j<=2*i-1;i++)  printf("*");
```

(3) 准备在输出下一行的数据之前先换行一下：

```
printf("\n");
```

该算法的流程图如图4.4所示。

9．对以上各程序准备好测试数据。

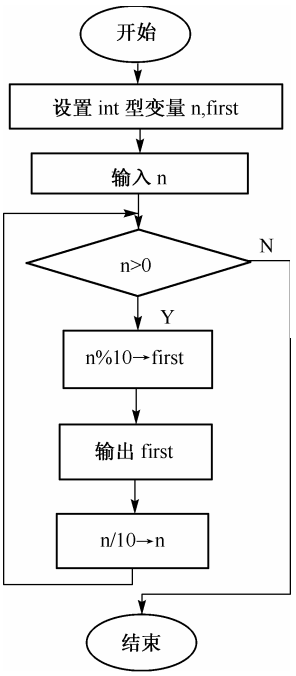

图 4.3　程序 2 流程图

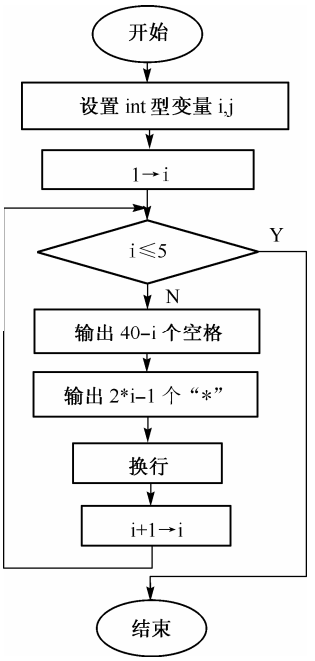

图 4.4　程序 4 流程图

三、实验内容

1．将验证题输入、编译、连接并运行，验证预习结果。

2．将未改正的改错题输入、编译后，抄下其中的编译错误信息。然后逐一改正错误，直至没有编译错误，再连接后运行。注意如何输入一批数据，及输入数据的结束方式，将实验数据记录下来。

3．将已编写好的程序 1 输入、编译、连接并运行。

4．将已编写好的程序 2 输入、编译、连接并运行，用不同的数据测试其执行结果。

5．将已编写好的程序 3 输入、编译、连接并运行，用不同的数据测试其执行结果。

6．将已编写好的程序 4 输入、编译、连接并运行。

四、实验报告要求

1．分析验证题的运行结果。

2．将改错题的正确程序清单写下来，并加上注释。写出试验数据及运行结果。并回答：这些输入的成绩，全部保存下来没有？能否实现输出所有高于平均分的成绩？

3．列出上机运行后的程序 1 清单，画出程序流程图。

4．列出上机运行后的程序 2 清单，对关键语句加上必要的注释。

5．列出上机运行后的程序 3 清单，并画出程序流程图。

6．列出上机运行后的程序 4 清单，对关键语句加上必要的注释。

实验 5　函　　数

一、实验目的

1．掌握函数的定义格式。

2．掌握函数的调用过程，深刻理解函数调用时形实结合“值传递”过程。

3．理解函数的嵌套调用和递归调用。

4．熟悉程序调试工具的使用，掌握跟踪调试程序的执行过程。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 4 章的内容：

① 函数的概念；

② 函数的定义格式、声明格式、调用规则及调用过程中数据传递方式；

③ 函数的嵌套调用和递归调用。

2．阅读有关单步跟踪功能键的使用，了解它们的区别。

3．验证题 1：阅读下列程序，写出预习结果。

```
#include <stdio.h>
void main()
{   void   fun(int i, int j, int k);    /* 函数的申明 */
    int x,y,z;
    x=y=z=6;
```

```
    fun(x,y,z);                          /* 函数的调用，没有返回值的函数调用 */
    printf("x=%d;y=%d;z=%d\n",x,y,z);
}
void  fun(int i, int j, int k)           /* 函数的定义 */
{   int t;
    t=(i+j+k)*2;
    printf("t=%d\n",t);
}
```

相关知识：函数调用的三种方式是

(1) 函数语句。适合调用 void 型函数或不需使用函数返回值的情况。

(2) 函数表达式。适合调用有函数返回值的情况。例：min_value =5*min(4,8);

(3) 函数参数。适合调用有函数返回值的情况。例：min_value = min(x, min(y, z));

4．验证题 2：阅读下列程序，写出预习结果。

```
#include <stdio.h>
void main()
{   int x=10,y=20;
    void  swap (int ,int);               /* 函数的申明 */
    printf("(1)in main :x=%d,y=%d\n",x,y);
    swap(x,y);                           /* 函数的调用，没有返回值的函数调用 */
    printf("(4)in main :x=%d,y=%d\n",x,y);
}
void swap (int m,int n)                  /* 函数的定义 */
{   int  temp;
    printf("(2)in main :m=%d,n=%d\n",m,n);
    temp=m;m=n;n=temp;
    printf("(3)in main :m=%d,n=%d\n",m,n);
}
```

相关知识：函数定义时，函数名后面圆括号内的参数称为形参；函数调用时，函数名后面圆括号内的表达式称为实参。对于形参，定义时仅仅只是指明它的类型，并不在内存中为其分配存储单元，只是在调用时才分配临时存储单元。当函数执行结束、返回主调函数时，该存储单元立即释放。形参具有“用之则建，用完则撤”的特点。

5．编写程序 1：编写一求 n!的函数。在主函数中输入 m 和 n，按下述公式计算并输出 C_m^n 的值。

$$C_m^n = \frac{m!}{n!(m-n)!}$$

提示：定义一个求阶乘的函数，然后在主函数中调用三次，分别求得 m!、n!和(m–n)!。。下面给出三种不同的阶乘求解方法，选择其中之一实现。

(1) $f(n)=n!=1\times2\times3\times\cdots\times n$

```
for(i=1,t=1;i<=n;i++)  t=t*i;
```

(2) $f(n)=n!=n\times(n-1)\times(n-2)\times\cdots\times1$

```
for(i=n,t=1;i>=1;i--)  t=t*i;
```

(3) 用递归来实现，递归函数如下：

$$f(n)=n!=\begin{cases}1 & n=1 \text{ 或 } n=0\\ n\times f(n-1) & n>1\end{cases}$$

6．编写程序 2：编写函数判断一个正整数 n 是否是素数，n 的值在主函数中输入。

7．验证题 3：阅读下列程序，假设 m 的值是 105，k 的值是 8，结果将会是多少？若 m 的值是 105，k 的值是 2，结果又会是多少？

```
#include <stdio.h>
void main()
{   int    m, k;
    void dtoo(int  n ,int r);       /* 函数的申明 */
    printf("Pleae input the decimal number:");
    scanf("%d",&m);
    printf("\nPlease input  a number  in (2,8):");
    scanf("%d",&k);
    dtoo(m,k);                      /* 函数的调用 */
}
void dtoo(int  n ,int r)            /* 函数的定义，定义了一个递归函数 */
{   if(n>=r)  dtoo(n/r,r);          /* 函数自己调用自己 */
    printf("%d",n%r);
}
```

8．对以上各程序准备好测试数据。

三、实验内容

1．将验证题 1 输入、编译、连接并运行，记录运行结果。再次运行程序，分别用单步跟踪键 F11 和 F10 调试该程序，理解它们的区别。如果将函数定义 fun 中的形参名 i, j, k 分别换成 x, y, z，则对程序有影响吗？试一下。

2. 将验证题 2 输入、编译、连接并运行，记录运行结果。注意，在 main 函数中调用 swap( ) 函数结束后，x 和 y 的值交换没有？

3．将编写好的程序 1 输入、编译、连接并运行。

4．将编写好的程序 2 输入、编译、连接并运行。

5．将验证题 3 输入、编译、连接并运行，验证预习结果。

四、实验报告要求

1．给出验证题 1 的实验结果。说说在 Turbo C 环境下用 F11 和 F10 单步跟踪该程序的不同之处。或者说说在 VC++环境下单步跟踪的集成操作，并回答：如果将函数定义 fun 中的形参名 i，j，k 分别换成 x，y，z，对程序有影响吗？为什么？

2．给出验证题 2 的实验结果。回答：调用 swap 函数后，x 和 y 的值交换没有？对结果进行分析。

3．将上机运行正确的程序 1 清单列出来，并加上必要的注释。

4．将上机运行正确的程序 2 清单列出来，并加上必要的注释。

5．分析验证题 3。

实验 6　数组与字符串

一、实验目的

1．掌握数组的定义、赋值及输入/输出。
2．掌握 C 语言中字符数组和字符串处理函数的使用。
3．熟悉 VC 集成环境下断点的设置。
4．熟悉常用的算法（如排序，有序数组的插入、删除等）。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 5 章的下述内容：
① 一维数组的输入/输出及常用算法；
② 二维数组的输入/输出及常用算法；
③ 字符数组的输入/输出及字符串处理函数的使用，如 strcpy( ), strcmp( ), strlen( ), strcat( )，等等；
④ int 型、float 型等的数组不能整体输入/输出，要逐个元素进行；而 char 型数组可以用 %s 一次性地实现输入/输出；
⑤ 任何类型的数组都不能整体地被赋值，必须逐个元素地进行。
2．仔细阅读 2.2.1 节或 3.4.2 节，熟悉程序调试中关于断点设置的内容。
3．改错题：下列程序的功能是输入 10 个整数，然后按每行 3 个数输出这些整数，最后输出其平均值。程序有错，已在注释中给出了错误提示，请改正。

```
#include <stdio.h>
#define  N  10
void main( )
{   int i,a[N];  float  av;
    for(i=0; i<N; i++)
        scanf("%d",a[i]);        /*对数组元素逐个进行输入,scanf 要求输入项为地址量 */
    for(i=0; i<N; i++)
        { printf("%d",a[i]);     /* 输出项之间没有分隔符，会紧凑在一起 */
          if(i%3==0)             /* 没有实现 3 个一行，第一行只有一个数 */
              printf("\n");
        }
    for(i=0;  i!=N;  i++)
        av+=a[i];                /* 累加器在累加之前清零，否则累加结果将不正确 */
    printf("av=%f\n",av/N);
}
```

4．编写程序 1：从键盘上输入若干个学生的成绩，以负数作为结束标志，计算出平均成绩，并输出低于平均分的学生成绩。算法流程如图4.5所示。
5．编写程序 2：编写程序，实现输入 10 个整数，将它们降序排列；然后再输入一个整数，并将其插入到该有序数列中，使数列仍保持降序。

提示:

(1) 输入 10 个整数。

(2) 用教材上介绍的选择法或冒泡法进行排序，注意是降序。

(3) 将一个整数插入到这个有序数列中的算法如下:

① 输入任意一个整数 k，如 scanf("%d", &k);

② 在数列中为整数 k 查找合适的插入位置 j: 将整数 k 和数列的元素从左到右逐个进行比较，若 k 大于某个 a[j]，则插入的位置就是 j。其实现过程如图4.6所示。

实现语句是

```
j=0;
while(a[j]>=k && j<10)   j++;
```

③ 由于数组是一种紧凑结构,要将插入位置后的(包括插入位置上的)元素依次往后移动一个位置，将 j 位置腾空出来。其实现过程如图4.7所示。

实现语句是

```
for(i=9;i>=j;i-- )   a[i+1]=a[i];
```

④ 将整数 k 插入到位置 j 上，真正实现插入。其实现过程如图4.8所示。

实现语句是

```
a[j]=k;
```

插入数据函数的算法流程如图 4.9 所示。部分源代码如下，请编写出各用户定义的函数，并使用断点调试技术来对每个函数的实现进行调试。

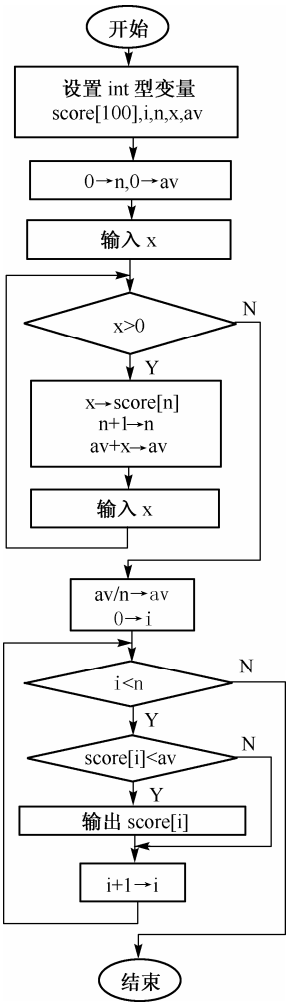

图 4.5　程序 1 流程图

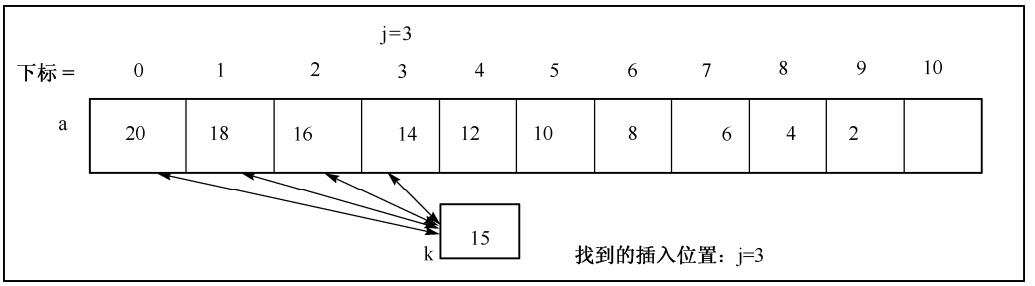

图 4.6　查找插入点

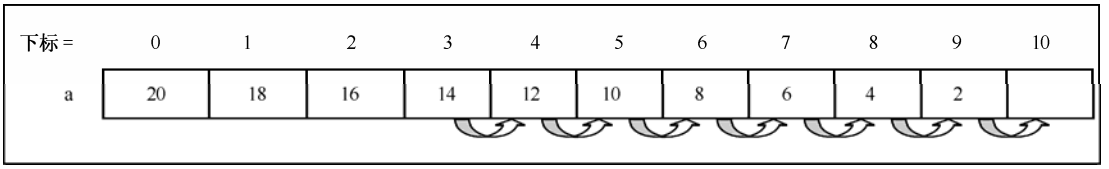

图 4.7　将下标为 j~9 的数组元素向后移动一个位置，腾空 j 号数组元素

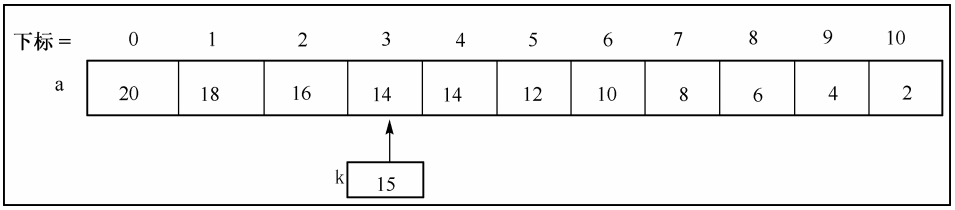

图 4.8　将 k 值插入到 j 号元素位置中去

```
#include  <stdio.h>
#define N 11                        /* 要定义 11 个元素的存储空间，给待插入的
                                       数据预留存储空间 */
void input(int [],int );            /* 声明输入数据函数 */
void sort(int [],int);              /* 声明排序函数 */
void insert(int [],int ,int);       /* 声明插入数据函数 */
void output(int [] ,int);           /* 声明输出数据函数 */
void main()
{   int a[N],k;
    input(a,10);                    /* 输入 10 个整数 */
    sort(a,10);                     /* 将这 10 个整数排序 */
    output(a,10);                   /* 输出已排序的 10 个数据 */
    printf("please input one integer");
    scanf("%d",&k);                 /* 输入一个整数 */
    insert(a, 10, k);               /* 将 k 插入到有序的数组
                                       中，保持原次序 */
    output(a,11);                   /* 输出已经插入数据的
                                       有序数列 */
}
void input(int a[],int n )          /* 定义输入数据函数 */
{……}
void sort(int a[], int n)           /* 定义排序函数 */
{……}
void insert(int a[],int n ,int k)/* 定义插入元素函数 */
{……}
void output(int a[] ,int n)         /* 定义输出数据函数 */
{……}
```

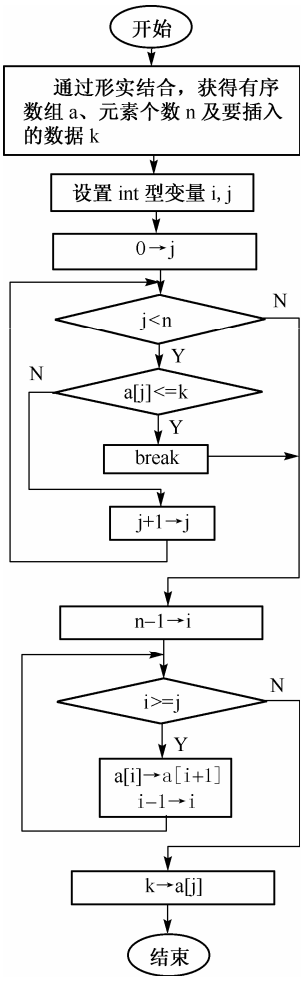

图 4.9　程序 2 插入数据函数 insert 流程图

6. 编写程序 3：任给一个 3 行 4 列的矩阵，编程实现求其中最大元素的值，以及所在的行号与列号。

7. 验证题：预习下列程序，若 s1 串为“country”，s2 串为“side”，结果如何？

```
#include  <stdio.h>
#include  <string.h>
void main( )
{   char   s1[80], s2[40];
    int    i = 0,  j = 0;
    printf(" \n Please input string1:");
    scanf("%s", s1);
    printf(" \n Please input string2:");
    scanf("%s", s2);
    while (s1[i]!= '\0' )           /* 在该处设置一个断点，对后面的语句单步跟踪调试 */
            i++;
```

```
    while (s2[j]!= '\0' )
            s1[i++]=s2[j ++];
    s1[i]= '\0';
    printf("\n New string: %s", s1);
}
```

8．填空题：下列程序的功能是统计从终端输入字符串中每个大写字母的个数。以#号结束输入。

```
#include  <stdio.h>
#include  <ctype.h>
void main ()
{   int num[26],i; char  c;          /*有 26 个大写字母，需要 26 个计数器 num[26]*/
    for (i=0;i<26;i++) num[i]=0;     /* 这 26 个计数器 num[26]在计数之前清零 */
    while (   (1)     != '#')        /* 统计从终端输入的大写字母个数 */
        if (isupper(c)) num[c-65]   (2)     ;
    for(i=0;i<26;i++)                /* 输出大写字母和该字母的统计结果 */
        if (num[i])printf("%c:%d\n", i   (3)     , num[i]);
}
```

9．编写程序 4：对输入的两个字符串进行比较，输出两个字符串中第一个不同字符的 ASCII 码之差。例如，输入的两个字符串分别为“abcdefg”和“abceef”，则对于这两个字符串，第 1 个不相同的字符是第 4 个位置上的’d’和’e’，输出结果为–1。

10．编写程序 5：输入字符串 s1、s2 及插入位置 f，在字符串 s1 中指定位置 f 后插入字符串 s2。例如，若输入 s1 为"BEIJING"、s2 为"123"和 f 为 3，则输出"BEI123JING"。

11．对以上各程序准备好测试数据。

三、实验内容

1．将未修改的改错题的原程序清单输入、编译、连接，注意在 VC 环境下此程序并没有编译或连接错误提示信息。但运行时，输入 10 个整数后有一错误窗口弹出，说明该程序有错误，逐一修改程序，直到输出结果符合要求为止。

2．将编写好的程序 1 输入、编译、连接并运行。

3．将编写好的程序 2 输入、编译、连接并运行。

4．将编写好的程序 3 输入、编译、连接并运行。

5．将验证题输入、编译、连接并运行，验证预习结果，利用断点设置技术跟踪执行程序。

6．将填空题输入、编译、连接并运行，输入不同的字符串，测试程序的功能。

7．将编写好的程序 4 输入、编译、连接并运行。

8．将编写好的程序 5 输入、编译、连接并运行。

四、实验报告要求

1．注意，改错题并没有编译错误，但运行时结果不对，写出体会和感想。

2．列出上机运行通过的程序 1 清单，并将实验数据及结果记录写出来。

3．列出上机运行通过的程序 2 清单。

4．列出上机运行通过的程序 3 清单，并画出程序流程图。

5．验证题的功能是什么？

6．将填空题的程序清单列出来，并给出两次输入不同字符串的结果。总结 getchar( )函数的使用方法。

7．列出上机运行通过的程序 4 清单，并画出程序流程图。

8．列出上机运行通过的程序 5 清单，并画出程序流程图。

实验 7　指　　针

一、实验目的

1．掌握指针变量的定义与引用。

2．掌握指针与数组的关系。

3．掌握用指针作为函数参数，进行“传地址”的过程。

4．掌握字符指针的使用。

5．了解指针数组的定义和使用。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 6 章的内容：

① 指针变量的定义、赋值及运算；

② 取地址运算符（&）和间接访问运算符（*）的运算规则；

③ 指向一维数组元素的指针；

④ 指向二维数组元素的指针，指向二维数组行的指针；

⑤ 指向字符串的指针和字符数组的区别；

⑥ 用指针作为函数的形参时，形实结合“传地址”的过程。

2．验证题 1：预习下列程序，写出预习结果。

```
#include <stdio.h>
void main()
{   int a,b,*pa,*pb;
    a=100;  b=500;  pa=&a;  pb=&b;    /* pa 存放变量 a 的地址，从而指向 a */
    printf("\na=%d\t*&a=%d\tb=%d\t*&b=%d",a, *&a, b, *&b);
    printf("\na=%d\t*pa=%d\tb=%d\t*pb=%d",a, *pa, b, *pb);
    printf("\n&a=%d\tpa=%d\t&b=%d\tpb=%d", &a, pa,&b,pb);
}
```

3．验证题 2：预习下列程序，写出预习结果。

```
#include <stdio.h>
void main()
{   int i,digit[]={1,2,3},*p;
    p=digit;                /* 将数组 digit 首地址赋给指针变量 p */
    printf("digit[i]\t* (digit+i)\tp[i]\t* (p+i)\n");
    for(i=0;i<3;i++)
        printf("%d\t\t%d\t\t%d\t%d\n",digit[i], *(digit+i),p[i],*(p+i));
}
```

相关知识：

(1) 指针指向数组的方法：将数组的首地址赋值给指针变量；

(2) 数组元素的表示方法：下标法，如 digit[i]形式；数组名地址法，如*(digit+i)；指针地址法，若指针变量 p=digit，则有*(p+i)；指针下标法，若指针变量 p=digit，则有 p[i]。

4．验证题 3：预习下列程序，写出预习结果。

```
#include <stdio.h>
void main( )
{   char  ch,*pc="C language program.",*p;
    printf("Enter a character:");
    scanf("%c",&ch);
    p=pc;
    while(*p!='\0' && *p!=ch) /* 指针所指向的字符不是空字符且不等于ch时，则向后查找 */
        p++;
    if(*p==ch)
        printf("The character %c is %dth\n",ch,p-pc+1);
                              /* 两指针相减求得所找字符距离串首的位置 */
    else
        printf("The character not found\n");
}
```

5．验证题 4：预习下列程序，写出预习结果。注意程序注释语句部分的提问。

```
#include <stdio.h>
void main( )
{   int a,b,c,*p=&c;
    void  sum(int ,int ,int *);
    void   swap1(int ,int);
    void   swap2(int *,int *);
    scanf("%d,%d",&a,&b);
    swap1(a,b);
    printf("a=%d,b=%d\n",a,b);      /* 此时 a 和 b 的值交换了没有? */
    swap2(&a,&b);
    printf("a=%d,b=%d\n",a,b);      /* 此时 a 和 b 的值交换了没有? */
    sum(a,b,p);
    printf("sum=%d\n",c);
}
void swap1(int  a,int b)
{   int t;
    t=a;    a=b;    b=t;
}
void swap2(int *a,int *b)
{   int t;
    t=*a;   *a=*b;  *b=t;
}
void  sum(int a,int b,int *c)       /* 该函数的功能是什么? */
{   *c=a+b;  }
```

相关知识：调用函数时注意数据在参数之间传递的是数值还是地址。

6．验证题 5：阅读下列程序，写出预习结果。

```
#include  <stdio.h>
void main()
```

```
{   char *a[]={"BASIC","FORTRAN","COBOL"};
    char **m;
    void f(char **);
    m=a;
    f(m);
}
void f(char **n)
{   printf("%s",*++n);
}
```

相关知识：① 多级指针的使用；② 多级指针与字符数组的操作。

7．填空题：下面程序的功能是利用指针实现从键盘输入 10 个整数，然后输出最小值。请填写实现。

```
#include <stdio.h>
void lookup(int *t,int *a,int n)
{   int k;
    *a=t[0];
    for(k=1;k<n;k++)
    if( ___(1)___ ) *a=t[k];   /* 如果 t[k]元素比*a 还小，则用 t[k]替换原来的*a */
}
void main()
{   int table[10];
    int k,min,*p=&min;
    for (k=0;k<10;k++)
        scanf("%d",table+k);            /* 注意 table+k 的使用 */
    lookup(__(2)___, __(3)___,10);      /* 函数调用，注意和形参的对应关系 */
    printf("min=%d\n",min);
}
```

8．编写程序1：利用指针编程实现找出一个字符串中最大的字符并把它放在最前面，其他字符往后顺序存放。例如，字符串"student"处理后成为"ustdent"。

9．编写程序 2：用指针方法编写一个函数，实现两个字符串的比较。

三、实验内容

1．将验证题 1 输入、编译、连接并运行，验证预习结果。

2．将验证题 2 输入、编译、连接并运行，验证预习结果。

3．将验证题 3 输入、编译、连接并运行，验证预习结果。

4．将验证题 4 输入、编译、连接并运行，验证预习结果。

5．将验证题 5 输入、编译、连接并运行，验证预习结果。

6．将填空题输入、编译、连接并运行。输入 10 个整数，看看能不能求出最小值。

7．将编写好的程序 1 输入、编译、连接并运行。

8．将编写好的程序 2 输入、编译、连接并运行。

四、实验报告要求

1．阅读验证题 1，回答如下问题：

① *&a 的结果是多少？说明*运算符和&运算符在一起时有什么特点？

② &a 和 pa 的值一样吗？pa 变量以%d 输出时的数据如何解释？

2．阅读验证题 2，回答如下问题：

若指针变量 p 的初值是 digit，则 digit[i], *(digit+i),p[i],*(p+i)这 4 种访问元素的结果一样吗？

3．阅读验证题 3，回答该程序的功能是什么？

4．阅读验证题 4，对调用 swap1 函数、swap2 函数及 sum 函数的过程进行分析。

5．阅读验证题 5，回答该程序的功能是什么？运行结果是什么？

6．将填空题 1 的程序清单列出来，并将上机实验的数据及结果写下来。

7．给出上机运行通过的程序 1 清单，并加上必要的注释。

8．给出上机运行通过的程序 2 清单，并加上必要的注释。

实验 8　C程序的模块化设计

一、实验目的

1．掌握全局变量和局部变量区别。

2．掌握静态局部变量的定义和特点。

3．理解宏的概念，掌握宏展开，特别是带参宏的展开。

4．了解文件包含的概念，掌握其用法。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 7 章的下述内容：

① 全局变量和局部变量的作用范围；

② 局部 static 型变量的特点；

③ 宏的概念、宏展开、带参数的宏和不带参数的宏；

④ 模块化程序设计方法。

2．阅读本书 2.1.6 节或 3.3 节有关多模块程序的编译、连接和运行操作方法。

3．验证题 1：阅读下列程序，写出预习结果。

```
# define  LOW  10
# define  LOW  10
#define  HIGH  5
#define  CHANGE  2
#include <stdio.h>
    int  i = LOW ;              /* 定义全局变量 i */
void main()
{   int workover(int  i) , reset(int i);
    int i=HIGH;                 /* 定义局部变量 i */
    reset(i/2);                 /* 考虑：调用结束后，i 的值改变了没有? */
    printf("(main)i=%d\n", i);
    reset(i=i/2);               /* 考虑：调用结束后，i 的值改变了没有? */
    printf("(main)i=%d\n", i);
    reset(i/2);
```

```
    printf("(main)i=%d\n", i);
    workover(i);
    printf("(main)i=%d\n", i);
}
int workover(int i)             /* i 是形式参数，也是局部变量 */
{   i=(i%i)*((i*i)/(2*i)+4) ;
    printf("(workover)i=%d\n", i);
    return(i);
}
int reset(int k)
{   i=k<=CHANGE ? HIGH:LOW ;    /* 此处 i 是全局变量 */
    printf("(reset)i=%d\n", i);
    return(i);
}
```

相关知识:

(1) 变量是存储空间的抽象，其特点是“用之不尽，一写就走”。在 main( )主函数中两个函数调用 reset(i/2)和 reset(i = i/2)，它们对主函数中的变量 i 的影响是不同的。第一个函数调用 reset( )中的 i 没有赋值，所以该函数调用结束后 i 还保持原来的值；而第二个 reset( )中的 i 被赋值了，所以函数调用结束后 i 的值会改变。

(2) 全局变量：在所有函数之外定义的变量，其有效范围从定义点到整个源程序结尾；局部变量：在函数内部定义的变量或在函数头部声明的形参，其有效范围只在所定义的函数内部，局部变量具有“用之则建，用完则撤”的特点。在不同函数内定义的变量同名时互不干扰。如果源程序存在局部变量和全局变量同名，则局部变量优先。

4．验证题 2：阅读下列程序，写出预习结果。

```
#include <stdio.h>
void main()
{   int i;
    int f(int);
    for (i=1; i<=5; i++)
    printf("(%d):% d\n", i,f(i));
    printf("\n");
}
int f(int n)
{   static int j=1;
    j=j*n;
    return(j);
}
```

相关知识：如果希望在函数调用结束后仍然保留函数局部变量的值，则可以将该局部变量定义为静态局部变量。静态局部变量具有如下一些特点:

(1) 全局寿命：静态局部变量占用静态存储区的存储单元。因此，在函数调用结束后，它的存储空间没有释放，它的值并不消失。这称为全局寿命。

(2) 局部可见性：其作用域只在定义它的函数内部，尽管它的值在函数调用结束后并不消失，但其他函数并不能访问它，只有进入到它所在的函数内部，它的值才可见。这称为局部可见性。

(3) 初始化的特点：在定义该局部变量时，若赋有初值，则初始化在编译时进行，只赋初值一次；若没有赋初值，则默认的初值为 0（数值型）或空字符（字符型）。

5．验证题 3：阅读下列程序，写出预习结果，注意程序运行时符号常量 NUM 的值是多少？

```
#include  "stdio.h"
#define  N  2
#define  M  N+1
#define  NUM  (M+1)*M/2                    /* 编译预处理命令 */
main()
 {  int i,n=0;
    for (i=1; i<=NUM; i++)
    {   n++;
        printf("%d\n",n);
    }
    printf("\n");
}
```

6．验证题 4：阅读下列程序，写出预习结果。

```
#define  POWER(x)  ((x)*(x))               /* 带参数的宏定义 */
#define  MAX(x,y)  (x)>(y)?(x):(y)
#define  PR  printf
#include <stdio.h>
void main()
{   int a,b,c,d,x;
    a=5;b=10;x=200;
    c=POWER(a+b);
    x=x/POWER(a+b);
    d=MAX(a+6,b);
    PR("c=%d,d=%d,x=%d \n", c,d,x);
}
```

7．选做：编写程序 1，求出两个整数的最大公约数、最小公倍数、两数的平方和及两数的平方差四种运算。该程序要求由五个模块（源程序文件）组成：

(1) 主控模块：包含主函数，完成两数的输入，并采用菜单方式让用户选择四种计算中的某一种，并能选择退出程序来结束运行。

(2) 子模块 1：求出两个整数的最大公约数。

(3) 子模块 2：求出两个整数的最小公倍数。

(4) 子模块 3：求出两个整数的平方和。

(5) 子模块 4：求出两个整数的平方差。

三、实验内容

1．将验证题 1 输入、编译、连接并运行，验证预习结果。

2．将验证题 2 输入、编译、连接并运行，验证预习结果。再次用 F11 单步跟踪该程序的运行，观察其中静态局部变量 j 在调用过程中的变化。

3．将验证题 3 输入、编译、连接并运行，验证预习结果。

4．将验证题 4 输入、编译、连接并运行，验证预习结果。然后，将程序的前两行改为

```
#define  POWER(x)  x*x
#define  MAX(x,y)  x>y?x:y
```

重新编译、连接程序，运行的结果是什么？

5．将编写好的程序 1 输入、编译、连接并运行。

四、实验报告要求

1．阅读验证题 1，分析每次函数调用结束后，主函数变量 i 的值。

2．阅读验证题 2，分析该程序的执行情况。

3．阅读验证题 3，分析符号常量 NUM 的值。

4．阅读验证题 4，分析程序的运行情况。若将程序的前两行改为

```
#define  POWER(x)  x*x
#define  MAX(x,y)  x>y?x:y
```

分析程序运行的结果。

5．列出经调试运行后，验证结果是正确的程序 1 多模块清单，说明你采用的模块化方法。

实验 9　构造数据类型

一、实验目的

1．掌握结构体、结构体类型的数组、指向结构体数据的指针的定义及使用方法。

2．掌握共用体数据类型、枚举数据类型的定义及使用方法。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 8 章的内容。

2．验证题 1：阅读下列程序，写出预习结果。

```
#include  <stdio.h>
void main()
{   union data                          /* 定义共用体类型的变量 student */
    {   int  num;
        float  score;
        char  *name;
    }student;
    student.score=78.5;                 /* 给 student 的成员变量 score 赋值 */
    student.name="liping";              /* 给 student 的成员变量 name 赋值 */
    student.num=1005;                   /* 给 student 的成员变量 num 赋值 */
    printf("%f\n", student.score);      /* 输出 student 的成员变量 score 的值 */
}
```

3．填空题：下列程序要求输出数组中三个学生的姓名，请填空将程序补充完整。

```
#include  <stdio.h>
struct student                          /* 定义结构体类型 student */
{   char num[7];
    char name[20];
    int age;
```

```
};
void  main()
{   struct student *p;                    /* 定义指向结构体数据的指针变量 p */
    struct student stud[3]= {{"200801","Liu",18},
                             {"200802","zhao",18},
                             {"200803","wang", 19}};
    for( (1) _;p<stud+3;p++)              /* 用循环语句输出三个学生的姓名 */
        printf("%s\n", (2));
}
```

相关知识:

(1) 指向结构体变量的指针定义的一般形式为

```
struct  结构体类型名   *指针变量名;
```

(2) 通过指向结构体变量的指针引用结构体成员的一般形式为

```
(*结构体指针变量).成员名   或   结构体指针变量->成员名
```

4. 验证题 2：阅读下列程序，写出预习结果。

```
#include  <stdio.h>
void  main( )
{
    enum  weekday
    {sun,mon,tue,wed,thu,fri,sat}day1,day2,day3;
        /* 定义 3 个枚举数据类型的变量 day1,day2,day3 */
    day1=sun;
        /* 给枚举数据类型的变量 day1 赋值 */
    day2=mon;
        /* 给枚举数据类型的变量 day2 赋值 */
    day3=tue;
        /* 给枚举数据类型的变量 day3 赋值 */
    printf("%d,%d,%d",day1,day2,day3);
        /* 以十进制整数的形式输出 3 个变量的值 */
}
```

相关知识:

(1) 枚举类型定义的一般形式为

```
enum  枚举类型名{枚举元素值表};
```

(2) 枚举元素作为常量是有值的，C 语言按定义时的顺序使它们的值为 0，1，2，…

5. 编写程序 1：有 10 个学生，每个学生的数据包括学号、姓名和三门课的成绩。从键盘输入 10 个学生的数据，要求打印出每个学生三门课的平均成绩，以及平均分最高的学生的所有数据（包括学号、姓名、三门课的成绩和平均分数）。

提示：用循环语句求出每个学生的平均分，然后再从这 10 个平均分中求出最高分，并记录该同学的元素下标，最后按要求输出结果。根据题意可定义如下结构及变量，算法流程如图4. 10所示。

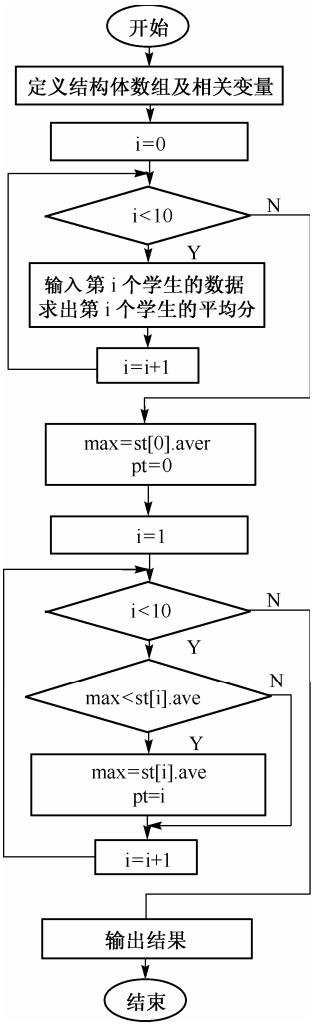

图 4.10　程序 1 的流程图

```
typedef  struct                    /* 定义结构体类型 STUD */
{   char  num[7];                  /* 字符数组 num 存储学号 */
    char  name[10];                /* 字符数组 name 存储姓名 */
    int   score[3];                /* 整型数组 score 存储三门课的成绩 */
    float aver;                    /* 单精度类型变量 aver 存储平均成绩*/
}STUD;
void  main( )
{   STUD  st[10];                  /*定义结构体STUD类型的数组,存储10个学生的数据 */
    int   i,j,pt;
    float max;
    …
    }
```

三、实验内容

1．改错题：下列程序要求输出结构体变量的学号、姓名及年龄，该程序存在错误，请找出错误并进行相应的修改。

```
#include  <stdio.h>
void main(  )
{   struct
    {   char  num[7];
        char  name[20];
        int  age;
    }student;
    student={"200801","liu qiang",18};  /* 要求给结构体变量 student 赋值 */
    printf("%s,%s,%d\n",student);       /*要求输出结构体变量的学号、姓名及年龄 */
}
```

相关知识:

(1) 不能将结构体变量作为一个整体进行输入或输出，一般都是对其成员变量逐个操作。结构体成员变量的引用形式为“结构体变量名.成员名”。

(2) 可以在定义结构体变量的同时对其赋值，称为结构体变量的初始化；其他情况下只能对结构体的成员变量逐个赋值。

2．将验证题 1 输入、编译、连接并运行，验证预习结果，并思考：程序运行后为什么不能得出正确的结果？

3．将填空题输入、编译、连接并运行。

4．将验证题 2 输入、编译、连接并运行，验证预习结果。

5．将编写好的程序 1 输入、编译、连接并运行多次，用不同的数据测试并记录输出结果。

四、实验报告要求

1．列出修改后的改错题的程序清单，并在程序修改处加上注释。

2．回答实验内容第 2 题的问题：程序运行后为什么不能得出正确结果？

3．列出填空题的程序清单。

4．列出程序 1 的清单，并加上必要的注释。

实验 10　位　运　算

一、实验目的

1．了解位运算符的作用及使用方法。

2．了解位段的概念及使用。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 9 章的内容。

2．填空题 1：下列程序要取一个整型变量 a 从右端开始的 2 ~ 5 位（从 0 位算起）的数据，请填空将程序补充完整。

```
#include  <stdio.h>
void  main( )
{
    unsigned  a,b,c,d;                          /* 定义无符号整型变量 a,b,c,d */
    scanf("%d",&a);                             /* 输入变量 a 的值 */
    b=a>>__(1)__;                               /* 将变量 a 右移若干位再赋值给 b，
                                                   使 a 右端的 2~5 位变成 b 的低 4 位 */
    c=~(~0<<_(2)_);                             /* 将 0 按位取反后左移若干位再按位取反，
                                                   使 c 低 4 位为 1，其余位为 0 */
    d=b&c;                                      /* 运算结果 d 得到 b 的低 4 位，
                                                   即 a 右端的 2 ~5 位 */
    printf("%o,%d\n%o,%d\n",a,a,d,d);           /* 以八进制和十进制两种形式输出 a,
                                                   b 的值 */
}
```

相关知识:

(1) “<<” 的运算规则是将运算对象的全部二进制位往左移动若干位，每左移一位，右边最低位添 0。

(2) “>>” 的运算规则是将运算对象的全部二进制位往右移动若干位，若运算对象最高位为 0，则每右移一位，左边最高位添 0；若运算对象最高位为 1，则每右移一位，最高位添 0 称为 “逻辑右移”，最高位添 1 称为 “算术右移”。

3．填空题 2：若某端口采集到的数据存放在 int 型的变量 a 中，其中变量 a 的第 5 位代表外部一个开关状态（1 为通，0 为断）。请填空完成下面程序，测试该开关的状态。

```
#include  <stdio.h>
void  main( )
{   int a;
    scanf ("%d",&a);     /* 输入变量 a 的值 */
    if( (1) )            /* 将 a 进行某种运算，当第 5 位的值为 1 时，运算结果要为真 */
    printf("switch is on!");
    else                 /* 当第 5 位的值为 0 时，运算结果要为假 */
    printf("switch is off!");
}
```

4．填空题 3：某端口采集到的数据存放在 int 型的变量 a 中，要求下列程序能使变量 a 的低 4 位翻转（即 1 变为 0，0 变为 1），其余位不变。请填空将程序补充完整。

```
#include <stdio.h>
void main( )
{
    int a,b;
    scanf("%d",&a);                          /* 输入变量 a 的值 */
    b=(1)                                    /* 将 a 进行某种运算，使 a 的低 4 位翻转，
                                                其余位不变 */
    printf("%d,%o\n%d,%o\n",a,a,b,b);        /* 以十进制和八进制两种形式输出 a,b 的值 */
}
```

5．验证题：阅读下列程序，写出预习结果。

```
#include <stdio.h>
struct packed_data      /* 定义位段型结构体变量 data */
{
    unsigned a : 3;     /* 成员 a 占 3 个 bit */
    unsigned b : 4;     /* 成员 b 占 4 个 bit */
    unsigned   : 2;         /* 第 6 行 */
    unsigned c : 5;     /* 成员 c 占 5 个 bit */
    int i;
}data;
void main()
{
    data.a = 4;         /* 给变量 data 的成员 a 赋值 */
    data.b = 10;        /* 给变量 data 的成员 b 赋值 */
    data.c = 45;        /* 给变量 data 的成员 c 赋值 */
    data.i = 25;        /* 给变量 data 的成员 i 赋值 */
    printf("%d,%o\n",data.c,data.c); /* 以十进制和八进制两种形式输出成员 c 的值 */
}
```

6．编写程序：编写一个程序，分别求出一个十六位二进制数的偶数位（即 D0，D2，…，D14）上的数据并输出。

三、实验内容

1．将填空题 1 输入、编译、连接并运行。

2．将填空题 2 输入、编译、连接并运行。

3．将填空题 3 输入、编译、连接并运行。

4．将验证题输入、编译、连接并运行，验证预习结果，并思考：程序第 6 行表示什么意思？程序运行后不能得出正确结果的原因是什么？

5．将编写好的程序输入、编译、连接并运行多次，用不同的数据测试并记录输出结果。

四、实验报告要求

1．给出填空题 1 的程序清单。

2．给出填空题 2 的程序清单。

3．给出填空题 3 的程序清单。

4．回答实验内容第 4 题的问题：程序第 6 行表示什么意思？程序运行后不能得出正确结果的原因是什么？

5．给出上机运行通过的程序清单，并加上必要的注释。

实验 11　文　　件

一、实验目的

1．掌握文件及缓冲文件系统、文件指针的概念。

2．学会使用文件打开、关闭、读、写等文件操作函数。

二、实验准备

1．复习教材《C 程序设计与应用教程》第 10 章的内容。

2．填空题 1：下列程序要求将字符串“Hello!”输出到当前目录下的“file1.txt”文件，请填空将程序补充完整。

```
#include <stdio.h>
void main( )
{   char a[20];          /* 定义字符数组 a */
    FILE *fpa;           /* 定义文件指针 fpa */
    scanf("%s",a);       /* 输入字符串“Hello!”保存在数组 a 中 */
    fpa=fopen( (1) );    /* 以“写”方式打开“file1.txt”文件 */
    fprintf( (2) );      /* 将数组 a 的数据格式化写入磁盘文件“file1.txt” */
    fclose(fpa);         /* 关闭文件“file1.txt” */
}
```

相关知识：

(1) fopen()函数：打开一个磁盘文件，一般调用格式为

文件指针= fopen（文件名，操作方式）；

(2) fclose()函数：关闭一个已打开的磁盘文件，一般调用格式为

fclose（文件指针）；

(3) fprintf()函数：实现将数据格式化写入磁盘文件，一般调用格式为

fprintf（文件指针，格式字符串，输出列表）；

3．填空题 2：下列程序要求将填空题 1 所建文件“file1.txt”的信息读出并输出到屏幕上，请填空完成要求的功能。

```
#include <stdio.h>
void main()
{   char b[20]="";       /* 定义字符数组 b，并初始化为空字符串 */
    FILE  *fpb;          /* 定义文件指针 fpb */
    fpb=fopen( (1) );    /* 以“读”方式打开“file1.txt”文件 */
    fscanf( (2) );       /* 将数据从文件“file1.txt”读出并存到数组 b */
    printf("%s", b);     /* 将数组 b 的数据输出到屏幕上 */
    fclose(fpb);         /* 关闭文件“file1.txt” */
}
```

相关知识:

fscanf()函数: 实现将数据从磁盘文件读出，一般调用格式为

```
fscanf（文件指针，格式字符串，输出列表）;
```

4．验证题：阅读下列程序，分析其功能。

```
#include <stdio.h>
#include <stdlib.h>
void  main( )
{   FILE  *in, *out;
    char  ch,infile[10],outfile[10];
    printf("The infile name:\n");
    scanf("%s",infile);
    printf("The outfile name:\n");
    scanf("%s",outfile);
    if((in=fopen(infile,"r"))==NULL)
    {   printf("Can not open infile!\n");
        exit(0);
    }
    if((out=fopen(outfile,"w"))==NULL)
    {   printf("Can not open outfile!\n");
        exit(0);
    }
    while(!feof(in))            /* 循环的条件是文件 in 没有结束 */
    {   ch= fgetc(in);
        fputc(ch,out ) ;
    }
    fclose(in);
    fclose(out);
}
```

相关知识:

(1) feof(fp): 用来测试 fp 所指文件的当前状态是否“文件结束”。如果是文件结束，函数值为 1（真）；否则函数值为 0（假）。

(2) fputc()函数: 把一个字符写到指定的磁盘文件上，一般调用格式为

```
fputc(ch ,fp);
```

(2) fgetc()函数: 从指定的文件读入一个字符，一般调用格式为

```
ch=fgetc(fp);
```

5．填空题 3：下列程序要求输入 3 名学生的信息（姓名、学号、成绩），并用 fwrite()函数将这些信息写入二进制文件“file3.dat”。请按要求填空将程序补充完整。

```
#include <stdio.h>
#include <stdlib.h>
struct student                  /* 定义结构体类型 student */
{   char  name[10];             /* 字符数组 name 存储姓名 */
    char  num[7];               /* 字符数组 num 存储学号 */
    int   score;                /* 整型变量 score 存储成绩 */
};
void main( )
{   struct student  stu[3];
    int i=0;
```

```
    FILE *fp=NULL;
    for(i=0;i<3;i++)            /* 通过循环输入 3 名学生信息 */
     (1)
    fp=fopen( (2) );            /* 用"写"方式打开二进制文件 */
    if(fp==NULL)
    {   printf("can't open this file\n");
        exit(0);
    }
    for(i=0;i<3;i++)            /* 用循环方式调用 fwrite()将信息写入文件 */
     (3)
    fclose(fp);
}
```

6．编写程序：编写一个程序，用 fread( )函数将填空题 3 所建文件“file3.dat”的数据读取出来，并显示到屏幕上。

三、实验内容

1．将填空题 1 输入、编译、连接并运行。
2．将填空题 2 输入、编译、连接并运行。
3．将验证题输入、编译、连接并运行，并思考：该程序的主要功能是什么？
4．将填空题 3 输入、编译、连接并运行。
5．将编写好的程序输入、编译、连接并运行。

四、实验报告要求

1．列出填空题 1 的程序清单。
2．写出填空题 2 的程序清单。
3．回答实验内容第 3 题的问题：该程序的主要功能是什么？
4．写出填空题 3 的完整程序。
5．写出上机运行后的程序清单，并加上必要的注释。

附录A　C语言编译常见错误信息列表

C 语言编译程序查出的源程序错误分为三类：严重错误、一般错误和警告。

(1) 严重错误（fatal error）：很少出现，它通常是内部编译出错。在发生严重错误时，编译立即停止，必须采取一些适当的措施并重新编译。

(2) 一般错误（error）：指程序的语法错误、磁盘或内存存取错误等。

(3) 警告（warning）：不阻止编译继续进行。它指出一些值得怀疑的情况，而这些情况本身又可以合理地作为源程序的一部分。

编译程序首先输出这三类出错信息，然后输出源文件名和发现出错的行号，最后输出信息的内容。请注意真正产生错误的行可能在指出行号的前一行或前几行。

下面按字母顺序分别列出 Turbo C 编译遇到的一些一般错误和警告信息。对每一条信息，均指出了可能产生的原因。有关 VC++ 6.0 中兼容 C 语言的出错信息，基本上与 Turbo C 编译相同，限于篇幅没有列出。

※一般错误

01 ‘×××’ not an argument

‘×××’不是函数参数。在源程序中将该标识符定义为一个函数参数，但此标识符没有在函数的参数表中出现。

02 Argument list syntax error

参数表出现语法错误。函数调用的一组参数之间必须以逗号隔开。

03 Array bound missin

数组的界限符“]”丢失。数组没有以一个右方括号结束。

04 Array size too large

定义的数组太大，可用内存不够。

05 Bad file name format in include directive

包含命令中文件格式不正确。包含文件名必须用引号（“filename.h”）或尖括号（<filename.h>）括起来，否则将产生此类错误。

06 Call of non-function

调用未定义函数。正被调用的函数无定义，通常是由于不正确的函数说明或函数名拼写错误引起的。

07 Cannot modify a constant object

不能修改一个常数对象。对定义为常量的对象进行不合法操作（例如，常量的赋值）将引起本错误。

08 Case outside of switch

case 出现在 switch 外。编译程序发现 case 语句出现在 switch 语句外面，通常是由于括号不配对引起的。

09　Case statement missing :

case 语句漏掉“:”。case 语句必须含有一个冒号终结的常量表达式。可能是丢了冒号或冒号前多了别的符号。

10　Case syntax error

case 语法错误。case 中有一些不正确的符号。

11　Character constant too long

字符常量太长。

12　Compound statement missing }

复合语句漏掉“}”。编译程序扫描到源文件结束时，未发现结束标记“}”，通常是由于花括号不配对引起的。

13　Constant expression required

要求常量表达式。数组的大小必须是常量。本错误通常由于#define 常量的拼写出错而引起。

14　Could not find file‘×××. ×××’

找不到‘×××. ×××’文件。编译程序找不到指定的文件。

15　Declaration needs type or storage class

说明必须给出类型或存储类。说明必须包含一个类型或一个存储类，如说明“i, j:”是不正确的。

16　Declaration syntax error

说明出现语法错误。某个说明丢失了某些符号或有多余的符号。

17　Default outside of switch

default 在 switch 外出现。编译程序发现 default 语句出现在 switch 语句之外，通常是由于括号不配对引起的。

18　Define directive needs an identifier

define 命令必须有一个标识符。#define 后面的第一个非空格符必须是一个标识符。若编译程序发现一些其他字符，则出现本错误。

19　Division by zero

除数为零。常量表达式中，出现除数为零的情况。

20　Do statement must have while

do 语句中必须有 while。程序中含有一条无 while 关键字的 do 语句。

21　Do-while statament missing (

do～while 语句中漏掉“（”。在 do 语句中，编译程序发现 while 关键字后无左括号。

22　Do-while statament missing)

do～while 语句中漏掉“）”。在 do 语句中，编译程序发现 while 关键字后无右括号。

23　Do-while statament missing ;

do～while 语句中漏掉“；”。在 do 语句中，编译程序发现 while 关键字后无分号。

24　Duplicate case

case 的情况值不唯一。switch 语句的每个 case 必须有一个唯一的常量表达式值。

25 Error writing output file

写输出文件错。通常是由于磁盘空间太满引起的，可能要删除一些不必要的文件，重新编译。

26 Expression syntax

表达式语法错。当编译程序分析一个表达式并发现一些严重错误时，出现本错误。通常是由于两个连续操作符、括号不配对或缺少括号，以及前一语句漏掉了分号等引起的。

27 Extra parameter in call to ×××

调用×××函数时出现多余参数。

28 File name too long

文件名太长。#include 命令给出的文件名太长，编译程序无法处理。

29 For statement missing (

for 语句中漏掉“(”。在 for 语句中，编译程序发现在控制表达式后缺少左括号。

30 For statement missing)

for 语句中漏掉“)”。在 for 语句中，编译程序发现在控制表达式后缺少右括号。

31 For statement missing ;

for 语句中漏掉“;”。在 for 语句中，编译程序发现在控制表达式后缺少分号。

32 Function call missing)

函数调用缺少“)”。函数调用的参数表有几种语法错误，如左括号漏掉或括号不配对。

33 Function definition out of place

函数定义位置错。函数定义不可出现在另一函数内。

34 Goto statement missing label

goto 语句缺少标号。在 goto 关键字后必须有一个标号。

35 If statement missing (

if 语句缺少“(”。在 if 语句中，if 关键字后面缺少左括号。

36 If statement missing)

if 语句缺少“)”。在 if 语句中，if 关键字后面缺少右括号。

37 Illegal character ‘c’(0xXX)

非法字符‘c’(0xXX)。编译程序发现输入文件有一些非法字符。

38 Illegal initialization

非法初始化。初始化必须是常量表达式，或是一个全局变量 extern, 或是 static 的地址加减一个常量。

39 Illegal octal digit

非法八进制数。编译程序发现一个八进制常数中包含了非八进制数字（例如，8 或 9）。

40 Illegal pointer subtraction

非法指针相减。这是由于试图以一个非指针变量减去一个指针变量而造成的。

41 Illegal structure operation

非法结构操作。结构只能使用（.）、取地址（&）和赋值（=）操作符，或作为函数的参数传递。当结构使用了其他操作符时，出现本错误。

42　Illegal use of floating point

非法浮点运算。浮点运算分量不允许出现在移位运算符、按位逻辑运算符、条件（？：）、间接（*）及其他一些运算符中。当上述运算符中使用了浮点运算分量时，出现本错误。

43　Illegal use of point

指针使用不合法。施于指针的运算符只能是加、减、赋值、比较、间接（*）或箭头。若用其他运算符，则出现本错误。

44　Improper use of a typedef symbol

typedef 符号使用不当。源文件中使用了一个 typedef 符号，符号变量应出现在一个表达式中。检查一下此符号的说明和可能的拼写错误。

45　Incompatible storage class

不相容的存储类。源文件的一个函数定义中使用了 extern 关键字，但只有 static（或根本没有存储类型）是允许的。

46　Incompatible type conversion

不相容的类型转换。源文件中试图把一种类型转换成另一种类型，但这两种类型是不相容的。例如，函数与非函数间转换、一种结构或数组与一种标准类型的转换、浮点数和指针间转换等。

47　Incorrect number format

不正确的数据格式。

48　Incorrect use of default

default 使用错。编译程序发现 default 关键字后缺少冒号。

49　Initialize syntax error

初始化语法错误。初始化过程缺少或多出了运算符，或者出现括号不匹配及其他不正常情况。

50　Invalid indirection

间接运算符错。间接运算符（*）要求非空指针作为运算分量。

51　Invalid macro argument separator

无效的宏参数分隔符。在宏定义中，参数必须用逗号分隔。当编译程序发现在参数名后面有其他非法字符时，出现本错误。

52　Invalid pointer addition

无效的指针相加。源程序中试图把两个指针相加。

53　Invalid use of arrow

箭头使用错。在箭头（->）运算符后必须跟一个标识符。

54　Invalid use of dot

点使用错。在点（.）运算符后必须跟一个标识符。

55　Lvalue required

左赋值请求。赋值运算符的左边必须是一个地址表达式，包括数值变量、指针变量、结构引用域、间接指针和数组分量。

56　Macro argument syntax error

宏参数语法错误。宏定义中的参数必须是一个标识符。

57 Macro ecpansion too long

宏扩展太长。一个宏扩展不能多于 4096 个字符。

58 Mismatch number of parameters in definition

函数定义中参数个数不匹配。函数定义中，参数和函数原型中提供的信息不匹配。

59 Misplaced break

break 位置错误。break 语句可能在 switch 语句或循环结构之外。

60 Misplaced continue

continue 位置错误。编译程序发现 continue 语句在循环结构之外。

61 Misplaced decimal point

十进制小数点位置错。浮点常数的指数部分有一个十进制小数点。

62 Misplaced else

else 位置错。else 语句缺少与之匹配的 if 语句。本错误的产生，除了由于 else 多余外，还有可能由于多余的分号或漏写了大括号及前面的 if 语句出现语法错误而引起。

63 Must take address of memory location

地址运算符&作用于不可编址的表达式。源文件中对不可编址的表达式使用了地址操作符（&），如对寄存器变量。

64 No file name ending

无文件名终止符。在#include 语句中，文件名缺少正确的闭引号（"）或右尖括号（>）。

65 Non-portable pointer assignment

不可操作指针赋值。源程序中将一个指针赋给一个非指针或相反。但作为特例，允许把常量零值赋给一个指针。

66 Non-portable pointer comparison

不可操作指针比较。源程序中将一个指针和一个非指针（常量零除外）进行比较。

67 Non-portable pointer conversion

不可操作返回类型转换。在返回语句中的表达式类型与函数说明中的类型不同。但如果函数的返回表达式是指针，则可以进行转换。此时，返回指针的函数可能送回一个常量零，而零被转换成一个适当的指针值。

68 Not an allowed type

不允许的类型。在源文件中说明了几种禁止的类型，如说明函数返回一个函数或数组。

69 Out of memory

内存不够。所有工作内存耗尽，应把文件放到一台有较大内存的机器去执行或简化源程序。

70 Pointer required on left side of ->

->操作符左边必须是一个指针。在->的左边未出现指针。

71 Redeclaration of ‘×××’

‘×××’重定义。此标识已经定义过。

72 Size of structure or array not known

结构或数组大小不确定。有些表达式（如 sizeof 或存储说明）中出现一个未定义的结构或一个空长度数组。如果结构长度不需要，那么在定义之前就可引用；如果数组不申请存储

空间或者初始化时给定了长度，那么就可以定义为空长。

73　Statement missing ;

语句缺少“;”。编译程序发现一个表达式语句后面没有分号。

74　Structure of union syntax error

结构或共用（联合）语法错误。编译程序发现在 struct 或 union 关键字后面没有标识符或左花括号{。

75　Structure size too large

结构太大。源文件中说明了一个结构所需的内存区域太大。

76　Subscripting missing]

下标缺少“]”。编译程序发现一个下标表达式缺少闭方括号。可能是由于漏掉、多写操作符或括号不匹配引起的。

77　Switch statement missing (

语句缺少“(”。在 switch 语句中，关键字 switch 后面缺少左括号。

78　Switch statement missing)

语句缺少“)”。在 switch 语句中，关键字 switch 后面缺少右括号。

79　Too few parameter in call to ‘×××’

调用‘×××’参数太少。调用指定的函数（该函数用一原型声明）时，给出的参数太少。

80　Too many default cases

default 情况太多。发现一个 switch 语句中有不止一个的 default 语句。

81　Too many initializers

初始化太多。

82　Too many storage classes in declaration

说明中存储类太多。一个说明只允许有一种存储类。

83　Too many types in declaration

说明中类型太多。一个说明只允许有一种下列基本类型：char、int、float、double、struct、union、enum 或 typedef。

84　Too much auto memory in function

函数中自动存储太多。当前函数说明的自动存储超过了可用的内存空间。

85　Too much code define in file

文件定义的代码太多。当前文件中函数的总长度超过 64 k 字节。可以移去不必要的代码或把源文件分开来写。

86　Too much global data define in file

文件中定义的全局数据太多。全局数据总数超过 64 k 字节。检查一些数组的定义是否太长。如果所有的说明都是必要的，那么考虑重新组织程序。

87　Type mismatch in parameter‘xxx’in call to‘yyy’

参数类型不匹配。

88　Type mismatch in redeclaration of‘×××’

重定义类型不匹配。源文件中把一个已经说明的变量重新说明为另一种类型。如果一个函数被调用，而后又被说明成非整型也会产生本错误。当发生这种情况时，必须在第一次调

用函数前给函数加上 extern 说明。

89 Unable to create output file‘×××.×××’

不能创建输出文件‘×××.×××’。当工作软盘已满或有写保护时产生本错误。如果软盘已满，则删除一些不必要的文件后重新编译；如果软盘有写保护，则把源文件移到一个可写的软盘上并重新编译。

90 Unable to create turboc.lnk

不能创建 turboc.lnk。编译程序不能创建临时文件，因为它不能存取磁盘或者磁盘已满。

91 Unable to open include file‘×××.×××’

不能打开包含文件‘×××.×××’。编译程序找不到该包含文件。

92 Unable to open input file‘×××.×××’

不能打开输入文件‘×××.×××’。当编译程序找不到源文件时出现本错误。检查文件名是否拼错，或检查对应的软盘或目录中是否有此文件。

93 Undefined label‘×××’

标号‘×××’未定义。函数中 goto 语句后的标号没有定义。

94 Undefined structure‘×××’

结构‘×××’未定义。源文件中使用了未经说明的某个结构。可能是由于结构名拼写错或缺少结构说明而引起。

95 Undefined symbol‘×××’

符号‘×××’未定义。通常是由于注释结束标志（*/）漏掉而引起。

96 Unterminated string or character constant

未终结的串或字符常量。编译程序发现串或字符常量开始后没有终结。

97 While statement missing (

while 的表达式语句漏掉“(”。在 while 语句中，关键字 while 的表达式后缺少左括号。

98 While statement missing ）

while 的表达式语句漏掉“)”。在 while 语句中，关键字 while 的表达式后缺少右括号。

※警告

01 ‘×××’declared but never used

说明了‘×××’但未使用。在源文件中说明了此变量，但没有使用。

02 ‘×××’is assigned a value which is never used

‘×××’被赋以一个不使用的值。此变量出现在一个赋值语句里，但直到函数结束都未使用过。

03 ‘×××’not part of structure

‘×××’不是结构的一部分。出现在点（.）或箭头（->）的左边的域名不是结构的一部分，或者点的左边不是结构，箭头的左边不指向结构。

04 Call to function with prototype

调用无原型的函数。

05 Code has no effect

代码无效。当编译程序遇到一个含有无效操作符的语句时发出本警告。例如，语句“a+b;”对每一个变量都不起作用，无需操作，且可能引起一个错误。

06　Conversion may lose significant digits

转换可能丢失高位数字。

07　Function should return a value

函数应该返回一个值。

08　No decaration for function‘×××’

函数‘×××’没有说明。

09　Non-portable pointer assignment

不可操作指针赋值。

10　Non-portable pointer comparison

不可操作指针比较。

11　Non-portable return type conversion

不可操作返回类型转换。

12　Parameter‘×××’is never used

参数‘×××’从未使用。

13　Possible use of‘×××’before definition

在定义‘×××’之前可能已使用。

14　Possible incorrect assignment

可能的不正确赋值。通常是由于把赋值号当做等号使用。

15　Superfluous & with function or array

在函数或数组中有多余的“&”号。

16　Undefined structure‘×××’

结构‘×××’未定义。在源文件中使用了该结构，但未定义。

17　Void function may not return a value

定义为 void 型的函数不可以有返回值。

18　Zero length structure

结构长度为零。在源文件中定义了一个总长度为零的结构。

附录B　补充练习题答案

第 1，2 章　C语言概述及数据类型与表达式

一、单项选择题

1．B　2．D　3．A　4．B　5．D　6．A　7．D　8．C　9．C
10．D　11．A　12．B　13．B　14．A　15．B　16．D　17．A　18．D
19．C　20．B　21．D　22．D　23．D　24．A　25．A　26．C　27．B
28．C　29．B　30．A

二、填空题

1．主
2．整数
3．{　}　定义　执行
4．八　十　十六
5．2　float　1
6．a=14
7．123.460 000
8．2，20.000 000
9．67，G
10．12
11．2　2　3　2　3　3　2　1
12．11，1，1
13．a=–3　b='a'　"end"
14．"%4lf%lf", &x, &y
15．11.50

第 3 章　算法的基本控制结构

一、单项选择题

1．B　2．C　3．C　4．D　5．A　6．B　7．C　8．A　9．D
10．B　11．B　12．D　13．D　14．A　15．D　16．A　17．B　18．B
19．C　20．A　21．D　22．C　23．D　24．D　25．A　26．C　27．C
28．A　29．B　30．D

二、填空题

1．z>10&& z<100||z<0
2．1

3．0
4．defABC
5．9866
6．x=4
7．a ： b　　c ： t1
8．k<=n　　k++
9．#* #* #*
10．sn/2　　sn+=hn
11．t*10　　s+t
12．j>0　　i
13．(a+b>c)&&(a+c>b)&&(b+c>a)
14．i<10　　j%3!=0
15．ch++　　printf("\n")

第 4 章　函数

一、单项选择题

1．C　2．D　3．A　4．D　5．D　6．D　7．B　8．D　9．B
10．C　11．C　12．A　13．C　14．C　15．A　16．C　17．D　18．D
19．B　20．A

二、填空题

1．void
2．9
3．return　0　　n*fac(n−1)
4．k%10　　a1==5||a2==5
5．显示主界面菜单　　退出系统，返回菜单
6．return　0　　return　1
7．n/16
8．x+8　　sin(x)
9．double　　return　s
10．x　　x*x+1

第 5 章　数组

一、单项选择题

1．B　2．D　3．B　4．B　5．A　6．C　7．B　8．D　9．A
10．C　11．C　12．D　13．A　14．B　15．B　16．C　17．B　18．D
19．C　20．C　21．B　22．A　23．B　24．C　25．B　26．C　27．C
28．C　29．B　30．B

二、填空题

1. p[i]=i; (s1+m-1)%i {s=1;continue;} j<i i
2. a[row][col]>max min>max
3. j<=I a[i][j]=a[j][i]
4. 6
5. abcbcc
6. x[0] x[0]
7. 3 7 15
8. (c=getchar()) ++或+=1 +65
9. 0 '\0' str1[i]-str2[i]
10. a a sum/n x[i]<ave
11. −5 10 20 40 −100 0 −50 60
    −100 −50 −5 0 10 20 40 60
12. * * * * *
    * * * * *
    * * * * *
    * * * * *
    * * * * *
13. 65
14. 1
    1 1
    1 2 1
    1 3 3 1
    1 4 6 4 1
    1 5 10 10 5 1
15. a[i−1] a[9−i]

第 6 章 指针

一、单项选择题

1. C 2. B 3. B 4. C 5. A 6. B 7. C 8. D 9. D
10. D 11. A 12. B 13. B 14. A 15. D 16. D 17. C 18. B
19. A 20. D 21. A 22. B 23. C 24. D 25. B 26. D 27. A
28. C 29. A 30. D

二、填空题

1. *s1 或 s1!='\0' *t0==*t1 或*t1==*t0 *s1++ *t1++
   s1=t0 或 s1=s1+strlen(str1)
2. c[k]=b[j] c[k]=a[i] a[i]=='\0' p=a+i strcat(c,p)

3．　k　　*k
4．　a[row][col]>max　　min>max
5．　*t　　*s--*t
6．　3　5
7．　*t++
8．　efgh
9．　str+strlen(str) –1　　t==0 或!t　　huiwen(str)
10．　efgh
11．　3*sizeof(double)
12．　1,2,3,4,9,8,7,6,5,0,
13．　Afternoon
14．　a
15．　8,7,6,5,4,3,2,1,

第 7 章　模块化程序设计

一、单项选择题

1．C　2．D　3．A　4．B　5．D　6．A　7．B　8．C　9．A
10．C　11．A　12．A　13．A　14．D　15．D　16．B　17．C　18．D
19．A　20．B　21．B　22．C　23．D　24．A　25．B　26．C　27．C
28．B　29．D　30．D

二、填空题

1．　编译　　内存　　传递
2．　运行（或执行）　　编译
3．　ar=9
4．　include　　包含
5．　# include　isodd(x)　((x)%2)
6．　宏定义　　文件包含　　条件编译
7．　# include <file1.h>
8．　定义形参的函数　　局部变量
9．　m=4　k=4
　　i=5　k=15
10．　15
11．　16　11
12．　2
　　4
　　6
13．　23，225
14．　8

20

12

15. #define N #endif

a[0] <

第 8 章 结构体

一、单项选择题

1. B 2. B 3. D 4. A 5. C 6. B 7. B 8. C 9. B
10. A 11. D 12. C 13. C 14. C 15. C 16. C 17. A 18. D
19. C 20. D

二、填空题

1.
```
/* 要求最高分数的学生数据，可以首先假设第一个学生的成绩最高，再使用一个
   for 循环把它与所有其他学生的成绩进行比较，找出最高的分数来。 */
fun( STU a[], STU *s)
{   int max,i;
    max=a[0].s;
    for( i=0;i<N; i++)
    if (a[i].s>max)
     {  max=a[i].s; *s=a[i];  }
}
```

2.
```
int fun (STREC *a, STREC *b)
{  int i,j=0,n=0,min;
  min=a[0].s;
  for(i=0;i<N;i++)        /* 先循环一次求出最低分数 */
    if(a[i].s<min) min=a[i].s;
       /* 再使用循环把所有成绩进行比较，找出所有的与最低分数
          相等的学生，每找出一个学生就放在数组 b 中。 */
       for(i=0;i<N;i++)
        if(min==a[i].s)
    {  *(b+j)=a[i];
       j++;   /* 所有的最低分数的学生的个数与数组 b 中最后的下标值一致。 */
      n++;
     }
  return n;
}
```

3.
```
void fun(  STREC *p  )
{   int i;
  double av=0;
  for(i=0;i<N;i++)
    av+=p->s[i];
  av/=N;
  p->ave=av;
}
```

4. 10x

5. ymd number+tab[lp][i] number

第 9 章　位运算

一、单项选择题

1．A　2．D　3．D　4．A　5．B　6．B　7．D　8．A　9．B
10．B　11．C　12．D　13．A　14．A　15．D

二、填空题

1．11110000
2．36
3．1
4．4
5．3

第 10 章　文件

一、单项选择题

1．B　2．B　3．A　4．D　5．C　6．C　7．D　8．D　9．C
10．B　11．B　12．D　13．A　14．A　15．C　16．B　17．D　18．C
19．C　20．A

二、填空题

1．"d1.dat","rb"
2．将文件 test1 的内容原样写到文件 test2 中
3．hello，
4．123　0
5．20 30
6．NULL　!feof(fp)　num++
7．"r"　c=fget(fp)　fclose(fp)
8．"bi.dat","w"或"bi.dat"，"w"或"bi．dat"，"W+t"
9．分析：

(1) 首先求得学生成绩的平均分。

(2) 然后设置一变量用于存放高于平均分的学生人数。

(3) 将所有学生的成绩数据依次与平均分进行比较，若高于平均分，则将变量值加 1。这样比较完成后变量中存放的即为分数高于平均分的学生人数。

```
double fun ( STREC *a, STREC *b, int *n )
{   double aver=0.0;
    int i,j=0;
    for(i=0;i<N;i++)
      aver+=a[i].s;
    aver/=N;
    for(i=0;i<N;i++)
```

```
    if(a[i].s>=aver)
    {  *(b+j)=a[i];
       j++;   }
  *n=j;
  return aver;
}
```

方法一：

```
  c=a/10*1000+a%10*10+b/10*100+b%10;
```

方法二：

```
  int d,e,n;
  *c=0; n=1;
  while(a > 0)
   {  d=a%10*10; e=b%10;
     *c+=(d+e)*n;
     n*=100;
    a=a/10; b=b/10;
   }
```

附录C　全国计算机等级考试二级C考试大纲

（2008 年最新版）

一、公共基础知识

（一）基本要求

1．掌握算法的基本概念。
2．掌握基本数据结构及其操作。
3．掌握基本排序和查找算法。
4．掌握逐步求精的结构化程序设计方法。
5．掌握软件工程的基本方法，具有初步应用相关技术进行软件开发的能力。
6．掌握数据的基本知识，了解关系数据库的设计。

（二）考试内容

1．基本数据结构与算法

(1) 算法的基本概念，算法复杂度的概念和意义（时间复杂度与空间复杂度）。

(2) 数据结构的定义，数据的逻辑结构与存储结构，数据结构的图形表示，线性结构与非线性结构的概念。

(3) 线性表的定义，线性表的顺序存储结构及其插入与删除运算。

(4) 栈和队列的定义，栈和队列的顺序存储结构及其基本运算。

(5) 线性单链表、双向链表与循环链表的结构及其基本运算。

(6) 树的基本概念，二叉树的定义及其存储结构，二叉树的前序、中序和后序遍历。

(7) 顺序查找与二分法查找算法，基本排序算法（交换类排序、选择类排序、插入类排序）。

2．程序设计基础

(1) 程序设计方法与风格。

(2) 结构化程序设计。

(3) 面向对象的程序设计方法，对象、方法、属性及继承与多态性。

3．软件工程基础

(1) 软件工程基本概念，软件生命周期概念，软件工具与软件开发环境。

(2) 结构化分析方法，数据流图，数据字典，软件需求规格说明书。

(3) 结构化设计方法，总体设计与详细设计。

(4) 软件测试的方法，白盒测试与黑盒测试，测试用例设计，软件测试的实施，单元测试、集成测试和系统测试。

(5) 程序的调试，静态调试与动态调试。

4．数据库设计基础

(1) 数据库的基本概念：数据库、数据库管理系统、数据库系统。

(2) 数据模型，实体联系模型及 E-R 图，从 E-R 图导出关系数据模型。

(3) 关系代数运算，包括集合运算及选择、投影、连接运算，数据库规范化理论。

(4) 数据库设计方法和步骤：需求分析、概念设计、逻辑设计和物理设计的相关策略。

（三）考试方式

1．公共基础的考试方式为笔试，与 C 语言（VisualBASIC、Visual FoxPro、Java、Access、Visual C++）的笔试部分合为一张试卷。公共基础部分占全卷的 30 分。

2．公共基础知识有 10 道选择题和 5 道填空题。

二、《C语言程序设计》基本要求

1．熟悉 Turbo C/Visual C++ 6.0 集成开发环境。

2．熟练掌握结构化程序设计的方法，具有良好的程序设计风格。

3．掌握程序设计中简单的数据结构和算法，并能阅读简单的程序。

4．在 Turbo C/Visual C++ 6.0 的集成环境下，能够编写简单的 C 程序，并具有基本的纠错和调试程序的能力。

（一）考试内容

1．C 语言的结构

(1) 程序的构成，main 函数和其他函数。

(2) 头文件、数据说明、函数的开始和结束标志，以及程序中的注释。

(3) 源程序的书写格式。

(4) C 语言的风格。

2．数据类型及其运算

(1) C 的数据类型（基本类型、构造类型、指针类型、无值类型）及其定义方法。

(2) C 运算符的种类、运算优先级和结合性。

(3) 不同类型数据间的转换与运算。

(4) C 表达式类型（赋值表达式、算术表达式、关系表达式、逻辑表达式、条件表达式、逗号表达式）和求值规则。

3．基本语句

(1) 表达式语句、空语句、复合语句。

(2) 输入/输出函数的调用，正确输入数据并正确设计输出格式。

4．选择结构程序设计

(1) 用 if 语句实现选择结构。

(2) 用 switch 语句实现多分支选择结构。

(3) 选择结构的嵌套。

5．循环结构程序设计

(1) for 循环结构。

(2) while 和 do～while 循环结构。

(3) continue 语句和 break 语句。

(4) 循环的嵌套。

6．数组的定义和引用

(1) 一维数组和二维数组的定义、初始化和数组元素的引用。

(2) 字符串与字符数组。

7．函数

(1) 库函数的正确调用。

(2) 函数的定义方法。

(3) 函数的类型和返回值。

(4) 形式参数与实在参数，参数值的传递。

(5) 函数的正确调用，嵌套调用、递归调用。

(6) 局部变量和全局变量。

(7) 变量的存储类别（自动、静态、寄存器、外部），变量的作用域和生存期。

8．编译预处理

(1) 宏定义和调用（不带参数的宏，带参数的宏）。

(2) “文件包含”处理。

9．指针

(1) 指针与指针变量的概念，指针与地址运算符。

(2) 变量、数组、字符串、函数、结构体的指针，以及指向变量、数组、字符串、函数、结构体的指针变量。通过指针引用以上各类型数据。

(3) 用指针作为函数参数。

(4) 返回指针值的指针函数。

(5) 指针数组，指向指针的指针，main 函数的命令行参数。

10．结构体（即“结构”）与共用体（即“联合”）

(1) 结构体和共用体类型数据的定义方法和成员的引用方法。

(2) 用指针和结构体构成链表，单向链表的建立，结点数据的输出、删除与插入。

(3) 用 typedef 说明一个新类型。

11．位运算

(1) 位运算符的含义及使用。

(2) 简单的位运算。

12．文件操作

只要求缓冲文件系统（即高级磁盘 I/O 系统），对非标准缓冲文件系统（即低级磁盘 I/O 系统）不要求。

(1) 文件类型指针（FILE 类型指针）。

(2) 文件的打开与关闭（fopen，fclose）。

(3) 文件的读写（fputc、fgetc、fputs、fgets、fread、fwrite、fprintf、fscanf 函数的应用），文件的定位（rewind 和 fseek 函数的应用）。

（二）考试方式

1．笔试：120 分钟，满分 100 分，其中含公共基础知识部分的 30 分。

2．上机：60 分钟，满分 100 分。

上机操作包括：(1) 填空；(2) 改错；(3) 编程。

参 考 文 献

[1] 蔡启先. C 语言程序设计教程题解与上机指导（第 2 版）. 重庆：重庆大学出版社，2003.

[2] 谭浩强. C 程序设计教程学习辅导. 北京：清华大学出版社，2007.

[3] 谭浩强. C++程序设计题解与上机指导. 北京：清华大学出版社，2005.

[4] 王功明等.计算机等级考试考点分析、题解与模拟——二级C语言（新大纲版）. 北京：电子工业出版社，2008.

[5] 蔡启先. C 语言程序设计教程（第 2 版）. 重庆：重庆大学出版社，2003.

[6] 谭浩强. C 程序设计教程. 北京：清华大学出版社，2007.

[7] （美）Schildt. H 著，王子恢，戴健鹏等译. C 语言大全（第四版）.北京：电子工业出版社，2001.

[8] 苏小红，陈惠鹏，孙志岗等. C 语言大学实用教程. 北京：电子工业出版社，2005.

[9] 秦维佳. C/C++程序设计实验教程. 北京：机械工业出版社，2007.

[10] 龙佑喜. C 语言程序设计上机指导与习题选解. 长沙：湖南教育出版社，2004.

[11] 郑平安，曾大亮等. 程序设计基础（C 语言）第二版. 北京：清华大学出版社，2006.

[12] K.N.King 著 吕秀锋译. C 语言程序设计现代方法. 北京：人民邮电出版社，2007.